

Global Caching for the Alternation-free μ -Calculus

Daniel Hausmann, Lutz Schröder, and Christoph Egger

Friedrich-Alexander Universität Erlangen-Nürnberg, Germany

Abstract

We present a sound, complete, and optimal single-pass tableau algorithm for the alternation-free μ -calculus. The algorithm supports global caching with intermediate propagation and runs in time $2^{\mathcal{O}(n)}$. In game-theoretic terms, our algorithm integrates the steps for constructing and solving the Büchi game arising from the input tableau into a single procedure; this is done on-the-fly, i.e. may terminate before the game has been fully constructed. This suggests a slogan to the effect that *global caching = game solving on-the-fly*. A prototypical implementation shows promising initial results.

1998 ACM Subject Classification F.4.1 Mathematical Logic - Temporal Logic

Keywords and phrases modal logic, fixpoint logic, satisfiability, global caching, coalgebraic logic

1 Introduction

The modal μ -calculus [24, 2] serves as an expressive temporal logic for the specification of sequential and concurrent systems containing many standard formalisms such as linear time temporal logic LTL [27, 32], CTL [6], and PDL [33]. Satisfiability checking in the modal μ -calculus is EXPTIME-complete [30, 9]. There appears to be, to date, no readily implementable reasoning algorithm for the μ -calculus, and in fact (prior to [22]) even for its fragment CTL, that is simultaneously *optimal*, i.e. runs in EXPTIME, and *single-pass*, i.e. avoids building an exponential-sized data structure in a first pass. Typical data structures used in worst-case-optimal algorithms are automata [9], games [12], and, for sublogics such as CTL, first-pass tableaux [8].

The term *global caching* describes a family of single-pass tableau algorithms [17, 20] that build graph-shaped tableaux bottom-up in so-called *expansion* steps, with no label ever generated twice, and attempt to terminate before the tableau is completely expanded by means of judicious intermediate *propagation* of satisfiability and/or unsatisfiability through partially expanded tableaux. Global caching offers wide room for heuristic optimization, regarding standard tableau optimizations as well as the order in which expansion and propagation steps are triggered, and has been shown to perform competitively in practice; see [20] for an evaluation of heuristics in global caching for the description logic *ALCT*. One major challenge with global caching algorithms is typically to prove soundness and completeness, which becomes harder in the presence of fixpoint operators. A global caching algorithm for PDL has been described by Goré and Widmann [19]; finding an optimal global caching algorithm even for CTL has been named as an open problem as late as 2014 [14] (a non-optimal, doubly exponential algorithm is known [14]).

The contribution of the present work is an optimal global-caching algorithm for satisfiability in the alternation-free μ -calculus, extending our earlier work on the single-variable (*flat*) fragment of the μ -calculus [22]. The algorithm actually works at the level of generality of the alternation-free fragment of the coalgebraic μ -calculus [5], and thus covers also logics beyond the realm of standard Kripke semantics such as alternating-time temporal logic ATL [1], neighbourhood-based logics such as the monotone μ -calculus that underlies

Parikh’s game logic [31], or probabilistic fixpoint logic. To aid readability, we phrase our results in terms of the relational μ -calculus, and discuss the coalgebraic generalization only at the end of Section 4. The model construction in the completeness proof yields models of size $2^{\mathcal{O}(n)}$.

We have implemented our algorithm as an extension of the Coalgebraic Ontology Logic Reasoner COOL, a generic reasoner for coalgebraic modal logics [21]; given the current state of the implementation of instance logics in COOL, this means that we effectively support alternation-free fragments of relational, monotone, and alternating-time [1] μ -calculi, thus in particular covering CTL and ATL. We have evaluated the tool in comparison with existing reasoners on benchmark formulas for CTL [18] (which appears to be the only candidate logic for which well-developed benchmarks are currently available) and on random formulas for ATL and the alternation-free relational μ -calculus, with promising results; details are discussed in Section 5.

Related Work The theoretical upper bound EXPTIME has been established for the full coalgebraic μ -calculus [5] (and earlier for instances such as the alternating-time μ -calculus AMC [35]), using a multi-pass algorithm that combines games and automata in a similar way as for the standard relational case, in particular involving the Safra construction. *Global caching* has been employed successfully for a variety of description logics [17, 20], and lifted to the level of generality of coalgebraic logics with global assumptions [15] and nominals [16].

A tableaux-based non-optimal (NEXPTIME) decision procedure for the full μ -calculus has been proposed in [23]. Friedmann and Lange [12] describe an optimal tableau method for the full μ -calculus that, unlike most other methods including the one we present here, makes do without requiring guardedness. Like earlier algorithms for the full μ -calculus, the algorithm constructs and solves a parity game, and in principle allows for an on-the-fly implementation. The models constructed in the completeness proof are asymptotically larger than ours, but presumably the proof can be adapted for the alternation-free case by using determinization of co-Büchi automata [28] instead of Safra’s determinization of Büchi automata [34] to yield models of size $2^{\mathcal{O}(n)}$, like ours. For non-relational instances of the coalgebraic μ -calculus, including the alternation-free fragment of the alternating-time μ -calculus AMC, the $2^{\mathcal{O}(n)}$ bound on model size appears to be new, with the best known bound for the alternation-free AMC being $2^{\mathcal{O}(n \log n)}$ [35].

In comparison to our own recent work [22], we move from the flat to the alternation-free fragment, which means essentially that fixpoints may now be defined by mutual recursion, and thus can express properties such as ‘all paths reach states satisfying p and q , respectively, in strict alternation until they eventually reach a state satisfying r ’. Technically, the main additional challenge is the more involved structure of eventualities and deferrals, which now need to be represented using cascaded sequences of unfoldings in the focusing approach; this affects mainly the soundness proof, which now needs to organize termination counters in a tree structure. While the alternation-free algorithm instantiates to the algorithm from [22] for flat input formulas, its completeness proof includes a new model construction which yields a bound of $3^n \in 2^{\mathcal{O}(n)}$ on model size, slightly improving upon the bound $n \cdot 4^n$ from [22]. We present the new algorithm in terms that are amenable to a game-theoretic perspective, emphasizing the correspondence between global caching and game-solving. In fact, it turns out that global caching algorithms effectively consist in an integration of the separate steps of typical game-based methods for the μ -calculus [12, 13, 30] into a single on-the-fly procedure that talks only about partially expanded tableau graphs, implicitly combining on-the-fly determinization of co-Büchi automata with on-the-fly solving of the resulting Büchi games [10]. This motivates the mentioned slogan that

global caching is on-the-fly determinization and game solving.

In particular, the propagation steps in the global caching pattern can be seen as solving an incomplete Büchi game that is built directly by the expansion steps, avoiding explicit determinization of co-Büchi automata analogously to [28]. One benefit of an explicit global caching algorithm integrating the pipeline from tableaux to game solving is the implementation freedom afforded by the global caching pattern, in which suitable heuristics can be used to trigger expansion and propagation steps in any order that looks promising.

2 Preliminaries: The μ -Calculus

We briefly recall the definition of the (relational) μ -calculus. We fix a set P of *propositions*, a set A of *actions*, and a set \mathfrak{V} of fixpoint variables. Formulas ϕ, ψ of the μ -calculus are then defined by the grammar

$$\psi, \phi ::= \perp \mid \top \mid p \mid \neg p \mid X \mid \psi \wedge \phi \mid \psi \vee \phi \mid \langle a \rangle \psi \mid [a] \psi \mid \mu X. \psi \mid \nu X. \psi$$

where $p \in P$, $a \in A$, and $X \in \mathfrak{V}$; we write $|\psi|$ for the size of a formula ψ . Throughout the paper, we use η to denote one of the fixpoint operators μ or ν . We refer to formulas of the form $\eta X. \psi$ as *fixpoint literals*, to formulas of the form $\langle a \rangle \psi$ or $[a] \psi$ as *modal literals*, and to p , $\neg p$ as *propositional literals*. The operators μ and ν *bind* their variables, inducing a standard notion of *free variables* in formulas. We denote the set of free variables of a formula ψ by $FV(\psi)$. A formula ψ is *closed* if $FV(\psi) = \emptyset$, and *open* otherwise. We write $\psi \leq \phi$ ($\psi < \phi$) to indicate that ψ is a (proper) subformula of ϕ . We say that ϕ *occurs free* in ψ if ϕ occurs as a subformula in ψ that is not in the scope of any fixpoint. Throughout, we *restrict to formulas that are guarded*, i.e. have at least one modal operator between any occurrence of a variable X and an enclosing binder ηX . (This is standard although possibly not without loss of generality [12].) Moreover we assume w.l.o.g. that input formulas are *clean*, i.e. all fixpoint variables are distinct, and *irredundant*, i.e. $X \in FV(\psi)$ for all subformulas $\eta X. \psi$.

Formulas are evaluated over *Kripke structures* $\mathcal{K} = (W, (R_a)_{a \in A}, \pi)$, consisting of a set W of *states*, a family $(R_a)_{a \in A}$ of relations $R_a \subseteq W \times W$, and a valuation $\pi : P \rightarrow \mathcal{P}(W)$ of the propositions. Given an *interpretation* $i : \mathfrak{V} \rightarrow \mathcal{P}(W)$ of the fixpoint variables, define $\llbracket \psi \rrbracket_i \subseteq W$ by the obvious clauses for Boolean operators and propositions, $\llbracket X \rrbracket_i = i(X)$, $\llbracket \langle a \rangle \psi \rrbracket_i = \{v \in W \mid \exists w \in R_a(v). w \in \llbracket \psi \rrbracket_i\}$, $\llbracket [a] \psi \rrbracket_i = \{v \in W \mid \forall w \in R_a(v). w \in \llbracket \psi \rrbracket_i\}$, $\llbracket \mu X. \psi \rrbracket_i = \mu \llbracket \psi \rrbracket_i^X$ and $\llbracket \nu X. \psi \rrbracket_i = \nu \llbracket \psi \rrbracket_i^X$, where $R_a(v) = \{w \in W \mid (v, w) \in R_a\}$, $\llbracket \psi \rrbracket_i^X(G) = \llbracket \psi \rrbracket_{i[X \mapsto G]}$, and μ, ν take least and greatest fixpoints of monotone functions, respectively. If ψ is closed, then $\llbracket \psi \rrbracket_i$ does not depend on i , so we just write $\llbracket \psi \rrbracket$. We write $x \models \psi$ for $x \in \llbracket \psi \rrbracket$. The *alternation-free fragment* of the μ -calculus is obtained by prohibiting formulas in which some subformula contains both a free ν -variable and a free μ -variable. E.g. $\mu X. \mu Y. (\Box X \wedge \Diamond Y \wedge \nu Z. \Diamond Z)$ is alternation-free but $\nu Z. \mu X. (\Box X \wedge \nu Y. (\Diamond Y \wedge \Diamond Z))$ is not. CTL is contained in the alternation-free fragment.

We have the standard *tableau rules* (each consisting of one *premise* and a possibly empty set of *conclusions*) which will be interpreted AND-OR style, i.e. to show satisfiability of a set of formulas Δ , it will be necessary to show that *every* rule application that matches Δ has *some* conclusion that is satisfiable. Our algorithm will use these rules in the expansion

step.

$$\begin{array}{ll}
(\perp) & \frac{\Gamma, \perp}{\Gamma, \perp} \\
(\wedge) & \frac{\Gamma, \psi \wedge \phi}{\Gamma, \psi, \phi} \\
(\langle a \rangle) & \frac{\Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \phi}{\psi_1, \dots, \psi_n, \phi} \\
(\neg) & \frac{\Gamma, p, \neg p}{\Gamma, p, \neg p} \\
(\vee) & \frac{\Gamma, \psi \vee \phi}{\Gamma, \psi \quad \Gamma, \phi} \\
(\eta) & \frac{\Gamma, \eta X. \psi}{\Gamma, \psi[X \mapsto \eta X. \psi]}
\end{array}$$

(for $a \in A$, $n \in \mathbb{N}$, $p \in P$); we refer to the set of modal rules $(\langle a \rangle)$ by \mathcal{R}_m and to the set of the remaining rules by \mathcal{R}_p and usually write rules with premise Γ and conclusion $\Sigma = \Gamma_1, \dots, \Gamma_n$ in sequential form, i.e. as (Γ/Σ) .

► **Example 1.** As our running example, we pick a non-flat formula, i.e. one that uses two recursion variables. Consider the alternation-free formulas

$$\psi_1 = \mu X. ((p \wedge (r \vee \Box \psi_2)) \vee (\neg q \wedge \Box X)) \quad \psi_2 = \mu Y. ((q \wedge (r \vee \Box X)) \vee (\neg p \wedge \Box Y))$$

(where $A = \{*\}$ and we write $\Box = [*]$, $\Diamond = \langle * \rangle$). The formulas ψ_1 and $\psi_2[X \mapsto \psi_1]$ state that all paths will visit p and q in strict alternation until r is eventually reached, starting with p and with q , respectively.

3 The Global Caching Algorithm

We proceed to describe our global caching algorithm for the alternation-free μ -calculus. First off, we need some syntactic notions regarding decomposition of fixpoint literals.

► **Definition 2 (Deferrals).** Given fixpoint literals $\chi_i = \eta X_i. \psi_i$, $i = 1, \dots, n$, we say that a substitution $\sigma = [X_1 \mapsto \chi_1]; \dots; [X_n \mapsto \chi_n]$ *sequentially unfolds* χ_n if $\chi_i <_f \chi_{i+1}$ for all $1 \leq i < n$, where we write $\psi <_f \eta X. \phi$ if $\psi \leq \phi$ and ψ is open and occurs free in ϕ (i.e. σ unfolds a nested sequence of fixpoints in χ_n innermost-first). We say that a formula χ is *irreducible* if for every substitution $[X_1 \mapsto \chi_1]; \dots; [X_n \mapsto \chi_n]$ that sequentially unfolds χ_n , we have that $\chi = \chi_1([X_2 \mapsto \chi_2]; \dots; [X_n \mapsto \chi_n])$ implies $n = 1$ (i.e. $\chi = \chi_1$). An *eventuality* is an irreducible closed least fixpoint literal. A formula ψ *belongs* to an eventuality θ_n , or is a θ_n -*deferral*, if $\psi = \alpha \sigma$ for some substitution $\sigma = [X_1 \mapsto \theta_1]; \dots; [X_n \mapsto \theta_n]$ that sequentially unfolds θ_n and some $\alpha <_f \theta_1$. We denote the set of θ_n -deferrals by $dfr(\theta_n)$.

E.g. the substitution $\sigma = [Y \mapsto \mu Y. (\Box X \wedge \Diamond Y)]; [X \mapsto \theta]$ sequentially unfolds the eventuality $\theta = \mu X. \mu Y. (\Box X \wedge \Diamond Y)$, and $(\Diamond Y)\sigma = \Diamond \mu Y. (\Box \theta \wedge \Diamond Y)$ is a θ -deferral. A fixpoint literal is irreducible if it is not an unfolding $\psi[X \mapsto \eta X. \psi]$ of a fixpoint literal $\eta X. \psi$; in particular, every clean irredundant fixpoint literal is irreducible.

► **Lemma 3.** *Each formula ψ belongs to at most one eventuality θ , and then $\theta \leq \psi$.*

► **Example 4.** Applying the tableau rules \mathcal{R}_m and \mathcal{R}_p to the formula $\psi_1 \wedge EG \neg r$, where ψ_1 is defined as in Example 1 and $EG \phi$ abbreviates $\nu X. (\phi \wedge \Diamond X)$, results in a cyclic graph, with relevant parts depicted as follows:

$$\begin{array}{c}
(\wedge) \frac{\psi_1 \wedge EG \neg r}{\psi_1, EG \neg r =: \Gamma_1} \\
(\vee, \wedge, \nu, \mu)^* \frac{\Gamma, p, \Box \psi_2[X \mapsto \psi_1]}{\psi_2[X \mapsto \psi_1], EG \neg r =: \Gamma_2} \quad \frac{\Gamma, \neg q, \Box \psi_1}{\Gamma_1} (\Diamond) \\
(\vee, \wedge, \nu, \mu)^* \frac{\Gamma, q, \Box \psi_1}{\Gamma_1} \quad \frac{\Gamma, \neg p, \Box \psi_2[X \mapsto \psi_1]}{\Gamma_2} (\Diamond)
\end{array}$$

where $\Gamma = \{-r, \Diamond EG \neg r\}$. The graph contains three cycles, all of which contain but never *finish* a formula that belongs to ψ_1 (where a formula belonging to an eventuality ψ_1 is said to be *finished* if it evolves to a formula that does not belong to ψ_1): In the rightmost cycle, the deferral $\delta_1 := \psi_1$ evolves to the deferral $\delta_2 := \Box\psi_1$ which then evolves back to δ_1 . For the cycle in the middle, δ_1 evolves to $\delta_3 := \Box\psi_2[X \mapsto \psi_1]$ which in turn evolves to $\delta_4 := \psi_2[X \mapsto \psi_1]$ before looping back to δ_3 . In the leftmost cycle, δ_1 evolves via δ_3 and δ_4 to δ_2 before cycling back to δ_1 . The satisfaction of ψ_1 is thus being postponed indefinitely, since $EG \neg r$ enforces the existence of a path on which r never holds. As a successful example, consider the graph that is obtained when attempting to show the satisfiability of $\psi_1 \wedge EG \neg q$, (where $\Gamma' := \{-q, \Diamond EG \neg q\}$):

$$\begin{array}{c}
(\wedge) \frac{\psi_2 \wedge EG \neg q}{\psi_2, EG \neg q =: \Gamma_3} \\
(\vee, \wedge, \mu, \nu)^* \frac{\Gamma', p, r \vee \Box\psi_2[X \mapsto \psi_1]}{\Gamma', p, r} \quad \frac{\Gamma', \Box\psi_1}{\Gamma_3} (\Diamond) \\
(\Diamond) \frac{\Gamma', p, r}{EG \neg q =: \Gamma_5} \quad \frac{\Gamma', p, \Box\psi_2[X \mapsto \psi_1]}{\psi_2[X \mapsto \psi_1], EG \neg q =: \Gamma_4} (\Diamond) \\
(\wedge, \nu) \frac{\Gamma'}{\Gamma_5} \quad (\Diamond) \frac{\Gamma', q, r \vee \Box\psi_1}{\Gamma_4} \quad \frac{\Gamma', \neg p, \Box\psi_2[X \mapsto \psi_1]}{\Gamma_4} (\Diamond) (\vee, \wedge, \mu)^*
\end{array}$$

The two loops through Γ_3 and Γ_4 are unsuccessful as they indefinitely postpone the satisfaction of the deferrals δ_2 and δ_3 , respectively; also there is the unsuccessful clashing node $\Gamma', q, r \vee \Box\psi_1$, containing both q and $\neg q$. However, the loop through Γ_5 is successful since it contains no deferral that is never finished; as all branching in this example is disjunctive, the single successful loop suffices to show that the initial node is successful. Our algorithm implements this check for ‘good’ and ‘bad’ loops by *simultaneously* tracking all deferrals that occur through the proof graph, checking whether each deferral is eventually finished.

We fix an input formula ψ_0 and denote the Fischer-Ladner closure [25] of ψ_0 by \mathbf{F} ; notice that $|\mathbf{F}| \leq |\psi_0|$. Let $\mathbf{N} = \mathcal{P}(\mathbf{F})$ be the set of all *nodes* and $\mathbf{S} \subseteq \mathbf{N}$ the set of all *state nodes*, i.e. nodes that contain only \top , non-clashing propositional literals (where p *clashes* with $\neg p$) and modal literals; so $|\mathbf{S}| \leq |\mathbf{N}| \leq 2^{|\psi_0|}$. Put

$$\mathbf{C} = \{(\Gamma, d) \in \mathbf{N} \times \mathcal{P}(\mathbf{F}) \mid d \subseteq \Gamma\}, \quad \text{and} \quad \mathbf{C}_G = \{(\Gamma, d) \in \mathbf{C} \mid \Gamma \in G\} \text{ for } G \subseteq \mathbf{N},$$

recalling that nodes are just sets of formulas; note $|\mathbf{C}| \leq 3^{|\psi_0|}$. Elements $v = (\Gamma, d) \in \mathbf{C}$ are called *focused nodes*, with *label* $l(v) = \Gamma$ and *focus* d . The idea of focusing single eventualities comes from work on LTL and CTL [26, 3]. In the alternation-free μ -calculus, eventualities may give rise to multiple deferrals so that one needs to focus *sets of deferrals* instead of single eventualities. Our algorithm incrementally builds a set of nodes but performs fixpoint computations on $\mathcal{P}(\mathbf{C})$, essentially computing winning regions of the corresponding Büchi game (with the target set of player 0 being the nodes with empty focus) on-the-fly.

► **Definition 5 (Conclusions).** For a node $\Gamma \in \mathbf{N}$ and a set \mathcal{S} of tableau rules, the set of *conclusions* of Γ under \mathcal{S} is

$$Cn(\mathcal{S}, \Gamma) = \{ \{\Gamma_1, \dots, \Gamma_n\} \in \mathcal{P}(\mathbf{N}) \mid (\Gamma/\Gamma_1 \dots \Gamma_n) \in \mathcal{S} \}.$$

We define $Cn(\Gamma)$ as $Cn(\mathcal{R}_m, \Gamma)$ if Γ is a state node and as $Cn(\mathcal{R}_p, \Gamma)$ otherwise. A set $N \subseteq \mathbf{N}$ of nodes is *fully expanded* if for each $\Gamma \in N$, $\bigcup Cn(\Gamma) \subseteq N$.

► **Definition 6 (Deferral tracking).** Given a node $\Gamma = \psi_1, \dots, \psi_n, \phi$ and a state node $\Delta \in \mathbf{S}$ that contains $[a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \phi$ as a subset, we say that Γ *inherits* ϕ from $(\langle a \rangle \phi, \Delta)$ and ψ_i from $([a]\psi_i, \Delta)$. For a non-state node $\Delta \in \mathbf{N}$, a node $\Gamma \in \mathbf{N}$ with $\phi \in \Gamma$, and $\psi \in \Delta$, Γ *inherits* ϕ from (ψ, Δ) if $\Gamma = \Gamma_i$ is conclusion of a non-modal rule $(\Gamma_0/\Gamma_1 \dots \Gamma_n)$ with

$\Gamma_0 = \Delta$ and either ψ has one of the forms ϕ , $\phi \vee \chi$, $\chi \vee \phi$, $\phi \wedge \chi$, $\chi \wedge \phi$, or $\psi = \eta X$. χ and $\phi = \chi[X \mapsto \psi]$. We put

$$\begin{aligned} Inh_m(\phi, \langle a \rangle \phi, \Delta) &= \{\Gamma \in \mathbf{N} \mid \Gamma \text{ inherits } \phi \text{ from } (\langle a \rangle \phi, \Delta)\} \\ Inh_m(\phi, [a] \phi, \Delta) &= \{\Gamma \in \mathbf{N} \mid \Gamma \text{ inherits } \phi \text{ from } ([a] \phi, \Delta)\} \\ Inh_p(\phi, \psi, \Delta) &= \{\Gamma \in \mathbf{N} \mid \Gamma \text{ inherits } \phi \text{ from } (\psi, \Delta)\}, \end{aligned}$$

where Δ is a state node in the first two clauses and a non-state node in the third clause. We write evs for the set of eventualities in \mathbf{F} . For a node $\Gamma \in \mathbf{N}$, the set of deferrals of Γ is

$$d(\Gamma) = \{\delta \in \Gamma \mid \exists \theta \in evs. \delta \in dfr(\theta)\}.$$

For a set $d \neq \emptyset$ of deferrals and nodes $\Gamma, \Delta \in \mathbf{N}$, we put

$$d_{\Delta \rightsquigarrow \Gamma} = \{\delta \in d(\Gamma) \mid \exists \theta \in evs. \exists \langle a \rangle \delta \in d. \Gamma \in Inh_m(\delta, \langle a \rangle \delta, \Delta) \text{ and } \delta, \langle a \rangle \delta \in dfr(\theta) \text{ or} \\ \exists [a] \delta \in d. \Gamma \in Inh_m(\delta, [a] \delta, \Delta) \text{ and } \delta, [a] \delta \in dfr(\theta)\}$$

if Δ is a state node, and

$$d_{\Delta \rightsquigarrow \Gamma} = \{\delta_1 \in d(\Gamma) \mid \exists \theta \in evs. \exists \delta_2 \in d. \Gamma \in Inh_p(\delta_1, \delta_2, \Delta) \text{ and } \delta_1, \delta_2 \in dfr(\theta)\}$$

if Δ is a non-state node. I.e. $d_{\Delta \rightsquigarrow \Gamma}$ is the set of deferrals that is obtained by *tracking* d from Δ to Γ , where Γ is the conclusion of a rule application to Δ . We put $\emptyset_{\Delta \rightsquigarrow \Gamma} = d(\Gamma)$, with the intuition that if the focus d is empty at (Δ, d) , then we *refocus*, i.e. choose as new focus for the conclusion Γ the set $d(\Gamma)$ of *all* deferrals in Γ .

► **Example 7.** Revisiting the proof graphs from Example 4, we fix additional abbreviations $\Gamma_6 := \Gamma, \neg p, \Box \psi_2[X \mapsto \psi_1]$, $\Gamma_7 := \Gamma', p, r \vee \Box \psi_2[X \mapsto \psi_1]$ and $\Gamma_8 := \Gamma', p, r$. In the first graph, e.g. $d(\Gamma_6) = \{\delta_3\}$ and $d(\Gamma_2) = \{\delta_4\}$; in the second graph, e.g. $d(\Gamma_7) = \{r \vee \Box \psi_2[X \mapsto \psi_1]\}$ and $d(\Gamma_8) = \emptyset$. In the first graph, the node Γ_6 inherits the deferral δ_3 from δ_4 at Γ_2 , i.e. $d(\Gamma_2)_{\Gamma_2 \rightsquigarrow \Gamma_6} = \{\delta_4\}_{\Gamma_2 \rightsquigarrow \Gamma_6} = \{\delta_3\}$ since $\Gamma_6 \in Inh_m(\psi_2[X \mapsto \psi_1], \Box \psi_2[X \mapsto \psi_1], \Gamma_2)$. Regarding the second graph, Γ_8 does not inherit any deferral from Γ_7 , i.e. $d(\Gamma_7)_{\Gamma_8 \rightsquigarrow \Gamma_7} = \{r \vee \Box \psi_2[X \mapsto \psi_1]\}_{\Gamma_8 \rightsquigarrow \Gamma_7} = \emptyset$ since $\Gamma_8 \in Inh_p(r, r \vee \Box \psi_2[X \mapsto \psi_1], \Gamma_7)$ but $r \vee \Box \psi_2[X \mapsto \psi_1] \in dfr(\psi_1)$ while $r \notin dfr(\psi_1)$, i.e. $r \vee \Box \psi_2[X \mapsto \psi_1]$ belongs to ψ_1 but r does not. This corresponds to the intuition that Γ_8 represents a branch originating from Γ_7 that actually finishes the deferral $r \vee \Box \psi_2[X \mapsto \psi_1]$.

We next introduce the functionals underlying the fixpoint computations for propagation of satisfiability and unsatisfiability.

► **Definition 8.** Let $C \subseteq \mathbf{C}$ be a set of focused nodes. We define the functions $f : \mathcal{P}(C) \rightarrow \mathcal{P}(C)$ and $g : \mathcal{P}(C) \rightarrow \mathcal{P}(C)$ by

$$\begin{aligned} f(Y) &= \{(\Delta, d) \in C \mid \forall \Sigma \in Cn(\Delta). \exists \Gamma \in \Sigma. (\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in Y\} \\ g(Y) &= \{(\Delta, d) \in C \mid \exists \Sigma \in Cn(\Delta). \forall \Gamma \in \Sigma. (\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in Y\} \end{aligned}$$

for $Y \subseteq C$. We refer to C as the *base set* of f and g .

That is, a focused node (Δ, d) is in $f(Y)$ if each rule matching Δ has a conclusion Γ such that $(\Gamma, d') \in Y$, where the focus d' is the set of deferrals obtained by tracking d from Δ to Γ .

► **Definition 9** (Proof transitionals). For $X \subseteq C \subseteq \mathbf{C}$, we define the *proof transitionals* $\hat{f}_X : \mathcal{P}(C) \rightarrow \mathcal{P}(C)$, $\hat{g}_X : \mathcal{P}(C) \rightarrow \mathcal{P}(C)$ by

$$\begin{aligned}\hat{f}_X(Y) &:= (f(Y) \cap \overline{F}) \cup (f(X) \cap F) = f(Y) \cup (f(X) \cap F) \\ \hat{g}_X(Y) &:= (g(Y) \cup F) \cap (g(X) \cup \overline{F}) = g(X) \cup (g(Y) \cap \overline{F}),\end{aligned}$$

for $Y \subseteq C$, where $F = \{(\Gamma, d) \in C \mid d = \emptyset\}$ and $\overline{F} = \{(\Gamma, d) \in C \mid d \neq \emptyset\}$ are the sets of focused nodes with empty and non-empty focus, respectively, and where C is the base set of f and g .

That is, $\hat{f}_X(Y)$ contains nodes with non-empty focus that have for each matching rule a successor node in Y as well as nodes with empty focus that have for each matching rule a successor node in X . The least fixpoint of \hat{f}_X thus consists of those nodes that finish their focus – by eventually reaching nodes from F with empty focus – and loop to X afterwards.

► **Lemma 10.** *The proof transitionals are monotone w.r.t. set inclusion, i.e. if $X' \subseteq X$, $Y' \subseteq Y$, then $\hat{f}_{X'}(Y') \subseteq \hat{f}_X(Y)$ and $\hat{g}_{X'}(Y') \subseteq \hat{g}_X(Y)$.*

► **Definition 11** (Propagation). For $G \subseteq \mathbf{N}$, we define $E_G, A_G \subseteq \mathbf{C}_G$ as

$$E_G = \nu X. \mu Y. \hat{f}_X(Y) \quad \text{and} \quad A_G = \mu X. \nu Y. \hat{g}_X(Y),$$

where \mathbf{C}_G is the base set of f and g .

Notice that in terms of games, the computation of E_G and A_G corresponds to solving an incomplete Büchi game. The set E_G contains nodes (Γ, d) for which player 0 has a strategy to enforce – for each infinite play starting at (Γ, d) – the Büchi condition that nodes in F , i.e. with empty focus, are visited infinitely often; similarly A_G is the winning region of player 1 in the corresponding game, i.e. contains the nodes for which player 1 has a strategy to enforce an infinite play that passes F only finitely often or a finite play that gets stuck in a winning position for player 1.

► **Example 12.** Returning to Example 4, we have $(\Gamma_1, d(\Gamma_1)) = (\Gamma_1, \{\psi_1\}) \in A_{G_1}$ and $(\Gamma_3, d(\Gamma_3)) = (\Gamma_3, \{\psi_1\}) \in E_{G_2}$ where G_1 and G_2 denote the set of all nodes of the first and the second proof graph, respectively; the global caching algorithm described later will therefore answer ‘unsatisfiable’ to Γ_1 , and ‘satisfiable’ to Γ_3 . To see $(\Gamma_1, \{\psi_1\}) \in A_{G_1}$ note that $A_{G_1} = \nu Y. \hat{g}_{A_{G_1}}(Y)$ by definition, so $A_{G_1} = (\hat{g}_{A_{G_1}})^n(\mathbf{C}_{G_1})$ for some n . For each focused node $(\Delta, d) \in \mathbf{C}_{G_1}$ there is a rule matching Δ all whose conclusions Γ satisfy $(\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in \mathbf{C}_{G_1}$, i.e. $g(\mathbf{C}_{G_1}) = \mathbf{C}_{G_1}$. Moreover, since all loops in G_1 indefinitely postpone some eventuality, no node with non-empty focus ever reaches one with empty focus, so $\hat{g}_\emptyset(\mathbf{C}_{G_1}) = \overline{F}$. Since \hat{g} is monotone and $(\Gamma_1, \{\psi_1\}) \in \overline{F}$, we obtain by induction over n that $(\Gamma_1, \{\psi_1\}) \in (\hat{g}_{A_{G_1}})^n(\mathbf{C}_{G_1})$. To see $(\Gamma_3, d(\Gamma_3)) = (\Gamma_3, \{\psi_1\}) \in E_{G_2}$, note that that starting from Γ_3 , the single deferral ψ_1 can be finished in finite time while staying in E_{G_2} . This holds because we can reach (Γ_8, \emptyset) by branching to the left twice and $(\Gamma_8, \emptyset) \in E_{G_2}$, since the loop through Γ_5 does not contain any deferrals whose satisfaction is postponed indefinitely and hence is contained in E_{G_2} .

► **Lemma 13.** *If $G' \subseteq G$, then $E_{G'} \subseteq E_G$ and $A_{G'} \subseteq A_G$.*

► **Lemma 14.** *Let $G \subseteq \mathbf{N}$ be fully expanded. Then $E_G = \overline{A_G}$.*

Our algorithm constructs a partial tableau, maintaining sets $G, U \subseteq \mathbf{N}$ of *expanded* and *unexpanded* nodes, respectively. It computes $E_G, A_G \subseteq \mathbf{C}_G$ in the propagation steps; as these sets grow monotonically, they can be computed incrementally.

Algorithm (Global caching). Decide satisfiability of a closed formula ϕ_0 .

1. (Initialization) Let $G := \emptyset$, $\Gamma_0 := \{\phi_0\}$, $U := \{\Gamma_0\}$.
2. (Expansion) Pick $t \in U$ and let $G := G \cup \{t\}$, $U := (U - \{t\}) \cup (\bigcup Cn(t) - G)$.
3. (Intermediate propagation) Optional: Compute E_G and/or A_G . If $(\Gamma_0, d(\Gamma_0)) \in E_G$, return ‘Yes’. If $(\Gamma_0, d(\Gamma_0)) \in A_G$, return ‘No’.
4. If $U \neq \emptyset$, continue with Step 2.
5. (Final propagation) Compute E_G . If $(\Gamma_0, d(\Gamma_0)) \in E_G$, return ‘Yes’, else ‘No’.

Note that in Step 5, G is fully expanded. For purposes of the soundness proof, we note an immediate consequence of Lemmas 13 and 14:

► **Lemma 15.** *If some run of the algorithm without intermediate propagation steps is successful on input ϕ_0 , then all runs on input ϕ_0 are successful.*

► **Remark.** For alternation-free fixpoint logics, the game-based approach (e.g. [13]) is to (1.) define a nondeterministic co-Büchi automaton of size $\mathcal{O}(n)$ that recognizes unsuccessful branches of the tableau. This automaton is then (2.) determinized to a deterministic co-Büchi automaton of size $2^{\mathcal{O}(n)}$ (avoiding the Safra construction using instead the method of [28]; here, alternation-freeness is crucial) and (3.) complemented to a deterministic Büchi automaton of the same size that recognizes successful branches of the tableau. A Büchi game is (4.) constructed as the product game of the carrier of the tableau and the carrier of the Büchi automaton. This game is of size $2^{\mathcal{O}(n)}$ and can be (5.) solved in time $2^{\mathcal{O}(n)}$.

Our global caching algorithm integrates analogues of items (1.) to (5.) in one go: We directly construct the Büchi game (thus replacing (1.) through (4.) by a single definition) step-by-step during the computation of the sets E and A of (un)successful nodes as nested fixpoints of the proof transitionals; the propagation step corresponds to (5.). Our algorithm allows for intermediate propagation, corresponding to solving the Büchi game on-the-fly, i.e. before it has been fully constructed.

4 Soundness, Completeness and Complexity

Soundness Let ϕ_0 be a satisfiable formula. By Lemma 15, it suffices to show that a run without intermediate propagation is successful.

► **Definition 16.** For a formula ψ , we define $\psi_X(\phi) = \psi[X \mapsto \phi]$, $\psi_X^0 = \perp$ and $\psi_X^{n+1} = \psi_X(\psi_X^n)$. We say that a Kripke structure \mathcal{K} is *stabilizing* if for each state x in \mathcal{K} , each $\mu X.\psi$, and each fixpoint-free context $c(-)$ such that $x \models c(\mu X.\psi)$, there is $n \geq 0$ such that $x \models c(\psi_X^n)$.

We note that finite Kripke structures are stabilizing and import the finite model property (without requiring a bound on model size) for the μ -calculus from [25]; for the rest of the section, we thus fix w.l.o.g. a stabilizing Kripke structure $\mathcal{K} = (W, (R_a)_{a \in A}, \pi)$ satisfying the target formula ϕ_0 in some state.

► **Definition 17 (Unfolding tree).** Given a formula ψ , an *unfolding tree* t for ψ consists of the syntax tree of ψ together with a natural number as additional label for each node that represents a least fixpoint operator. We denote this number by $t(\kappa, \mu X.\phi)$ for an occurrence of a fixpoint literal $\mu X.\phi$ at position $\kappa \in \{0, 1\}^*$ in ψ . We define the *unfolding* $\psi(t)$ of ψ according to an unfolding tree t for ψ by

$$X(t) = X \quad (\phi_1 \wedge \phi_2)(t) = \phi_1(t_1) \wedge \phi_2(t_2) \quad (\mu X.\phi_1)(t) = (\phi_1(t_1))_X^{t(\epsilon, \mu X.\phi_1)},$$

where t_i is the i -th child of the root of t , and similar clauses for $\langle a \rangle$, $[a]$, \vee , and ν as for \wedge .

Given a formula ψ , we define the order $<_\psi$ on unfolding trees for ψ by lexically ordering the lists of labels obtained by pre-order traversal of the syntax tree of ψ .

► **Definition 18 (Unfolding).** The *unfolding* of a formula ψ at a state x with $x \models \psi$ is defined as $unf(\psi, x) = \psi(t)$, where t is the least unfolding tree for ψ (w.r.t. $<_\psi$) such that $x \models \psi(t)$ (such a t exists by stabilization).

Note that in unfoldings, all least fixpoint literals $\mu X. \phi$ are replaced with finite iterates of ϕ .

► **Theorem 19 (Soundness).** *The algorithm returns ‘Yes’ on input ϕ_0 if ϕ_0 is satisfiable.*

Proof. (Sketch) We show that any node (Γ, d) that is constructed by the algorithm and whose label is satisfied at some state x in \mathcal{K} is successful, i.e. $(\Gamma, d) \in E_G$; the proof is by induction over the maximal modal depth of $unf(\delta, x)$ for $\delta \in d$. ◀

Completeness Assume that the algorithm answers ‘Yes’ on input ϕ_0 , having constructed the set $E := E_G$ of successful nodes. Put $D = \{(\Gamma, d) \in E \mid \Gamma \in \mathbf{S}\}$; note $|D| \leq |E| \leq 3^{|\phi_0|}$.

► **Definition 20 (Propositional entailment).** For a finite set Ψ of formulas, we write $\bigwedge \Psi$ for the conjunction of the elements of Ψ . We say that Ψ *propositionally entails* a formula ϕ (written $\Psi \vdash_{PL} \phi$) if $\bigwedge \Psi \rightarrow \phi$ is a propositional tautology, where modal literals are treated as propositional atoms and fixpoint literals $\eta X. \phi$ are unfolded to $\phi(\eta X. \phi)$ (recall that fixpoint operators are guarded).

► **Definition 21.** We denote the set of formulas in a node Γ that do *not* belong to an eventuality θ by

$$N(\Gamma, \theta) = \{\phi \in \Gamma \mid \phi \notin dfr(\theta)\}.$$

A set d of deferrals is *sufficient* for $\delta \in dfr(\theta)$ at a node Γ , in symbols $d \vdash_\Gamma \delta$, if $d \cup N(\Gamma, \theta) \vdash_{PL} \delta$. We write $\vdash_\Gamma \delta$ to abbreviate $\emptyset \vdash_\Gamma \delta$.

► **Definition 22 (Timed-out tableau).** Let $U \subseteq \mathbf{S} \times \mathbf{S}$ and let $L \subseteq U \times U$. We denote the set of L -successors of $v \in U$ by $L(v) = \{w \mid (v, w) \in L\}$. Let d be a set of deferrals. We put $to(\emptyset, n) = U$ for all n (*to* for *timeout*). For $d \neq \emptyset$, we put $to(d, 0) = \emptyset$ and define $to(d, m+1)$ to be the set of of focused nodes (Δ, d') such that writing $Cn(\Delta) = \{\Sigma_1, \dots, \Sigma_n\}$, we have $L(\Delta, d') = \{(\Gamma_1, d_1), \dots, (\Gamma_n, d_n)\}$ where for each i there exists $\Gamma \in \Sigma_i$ such that

- $\Gamma_i \vdash_{PL} \bigwedge \Gamma$ and $d_i \vdash_{\Gamma_i} d'_{\Delta \rightsquigarrow \Gamma}$, and
- $(\Gamma_i, d_i) \in to(d'', m)$ for some $d'' \subseteq d(\Gamma_i)$ with $d'' \vdash_{\Gamma_i} d_{\Delta \rightsquigarrow \Gamma}$.

If for each focused node $(\Gamma, d) \in U$ there is a number m such that $(\Gamma, d) \in to(d(\Gamma), m)$, then L is a *timed-out tableau* over U .

Roughly, $to(d, m)$ can be understood as the set of all focused nodes in U that finish all deferrals in d within m modal steps, i.e. with *time-out* m ; this is similar to Kozen’s μ -counters [24].

► **Lemma 23 (Tableau existence).** *There exists a timed-out tableau over D .*

Proof sketch. Since $D \subseteq E_G$, we can define $L \subseteq D \times D$ in such a way that all paths in L visit F (the set of nodes with empty focus) infinitely often, so every deferral contained in some node in D will be focused by the unavoidable eventual refocusing; this new focus will in turn eventually be finished so that L is a timed-out tableau. ◀

For the rest of the section, we fix a timed-out tableau L over D and define a Kripke structure $\mathcal{K} = (D, (R_a)_{a \in A}, \pi)$ by taking $R_a(v)$ to be the set of focused nodes in $L(v)$ whose label is the conclusion of an $(\langle a \rangle)$ -rule that matches $l(v)$ and by putting $\pi(p) = \{v \in D \mid p \in l(v)\}$.

► **Definition 24** (Pseudo-extension). The *pseudo-extension* $\widehat{\llbracket \phi \rrbracket}$ of ϕ in D is

$$\widehat{\llbracket \phi \rrbracket} = \{v \in D \mid l(v) \vdash_{PL} \phi\}.$$

► **Lemma 25** (Truth). In the Kripke structure \mathcal{K} , $\widehat{\llbracket \psi \rrbracket} \subseteq \llbracket \psi \rrbracket$ for all $\psi \in \mathbf{F}$.

Proof sketch. Induction on ψ , with an additional induction on time-outs in the case for least fixpoint literals, exploiting alternation-freeness. ◀

► **Corollary 26** (Completeness). If a run of the algorithm with input ϕ_0 returns ‘Yes’, then ϕ_0 is satisfiable.

Proof sketch. Combine the existence lemma and the truth lemma to obtain a model over D . Since $(\{\phi_0\}, d(\{\phi_0\})) \in E$ and $\widehat{\llbracket \phi_0 \rrbracket} \subseteq \llbracket \phi_0 \rrbracket$, there is a focused node in D that satisfies ϕ_0 . ◀

As a by-product, our model construction yields

► **Corollary 27.** Every satisfiable alternation-free fixpoint formula ϕ_0 has a model of size at most $3^{|\phi_0|}$.

Thus we recover the bound of $2^{\mathcal{O}(n)}$ for the alternation-free relational μ -calculus, which can be obtained, e.g., by carefully adapting results from [12] to the alternation-free case; for the alternation-free fragment of the alternating-time μ -calculus, covered by the coalgebraic generalization discussed next, the best previous bound appears to be $n^{\mathcal{O}(n)} = 2^{\mathcal{O}(n \log n)}$ [35].

Complexity Our algorithm has optimal complexity (given that the problem is known to be EXPTIME-hard):

► **Theorem 28.** The global caching algorithm decides the satisfiability problem of the alternation-free μ -calculus in EXPTIME, more precisely in time $2^{\mathcal{O}(n)}$.

The Alternation-Free Coalgebraic μ -Calculus Coalgebraic logic [5] serves as a unifying framework for modal logics beyond standard relational semantics, subsuming systems with, e.g., probabilistic, weighted, game-oriented, or preference-based behaviour under the concept of coalgebras for a set functor F . All our results lift to the level of generality of the (alternation-free) coalgebraic μ -calculus [4]; details are in a technical report at <https://www8.cs.fau.de/hausmann/afgc.pdf>. In consequence, our results apply also to the alternation-free fragments of the alternating-time μ -calculus [1], probabilistic fixpoint logics, and the monotone μ -calculus (the ambient fixpoint logic of Parikh’s game logic [31]), as all these can be cast as instances of the coalgebraic μ -calculus.

5 Implementation and Benchmarking

The global caching algorithm has been implemented as an extension of the *Coalgebraic Ontology Logic Reasoner* (COOL) [21], a generic reasoner for coalgebraic modal logics, available at <https://www8.cs.fau.de/research:software:cool>. COOL achieves its genericity by instantiating an abstract core reasoner that works for all coalgebraic logics to concrete instances of logics; our global caching algorithm extends this core. Instance logics implemented

in COOL currently include relational, monotone, and alternating-time logics, as well as any logics that arise as fusions thereof. In particular, this makes COOL, to our knowledge, the only implemented reasoner for the alternation-free fragment of the alternating-time μ -calculus (a tableau calculus for the sublogic ATL is prototypically implemented in the TATL reasoner [7]) and the star-nesting free fragment of Parikh’s game logic.

Although our tool supports the full alternation-free μ -calculus, we concentrate on CTL for experiments, as this appears to be the only candidate logic for which substantial sets of benchmark formulas are available [18]. CTL reasoners can be broadly classified as being either *top-down*, i.e. building graphs or tableaux by recursion over the formula, or *bottom-up*; the two groups perform very differently [18]. We compare our implementation with the top-down solvers TreeTab [14], GMUL [18], MLSolver [11] and the bottom-up solvers CTL-RP [36] and BDDCTL [18]. Out of the top-down solvers, only TreeTab is single-pass like COOL; however, TreeTab has suboptimal (doubly exponential) worst-case runtime. MLSolver supports the full μ -calculus. For MLSolver, CTL-RP and BDDCTL, formulas have first been *compacted* [18]. All tests have been executed on a system with Intel Core i7 3.60GHz CPU with 16GB RAM, and a stack limit of 512MB.

On the benchmark formulas of [18], COOL essentially performs similarly as the other top-down tools, and closer to the better tools when substantial differences show up. As an example, the runtimes of COOL, TreeTab, GMUL, MLSolver, CTL-RP, and BDDCTL on the Montali-formulas [29, 18] are shown in Figure 1. To single out one more example, Figure 2 shows the runtimes for the alternating bit protocol benchmark from [18]; COOL performs closer to GMUL than to MLSolver on these formulas.

This part of the evaluation may be summed up as saying that COOL performs well despite being, at the moment, essentially unoptimized: the only heuristics currently implemented is a simple-minded dependency of the frequency of intermediate propagation on the number of unexpanded nodes.

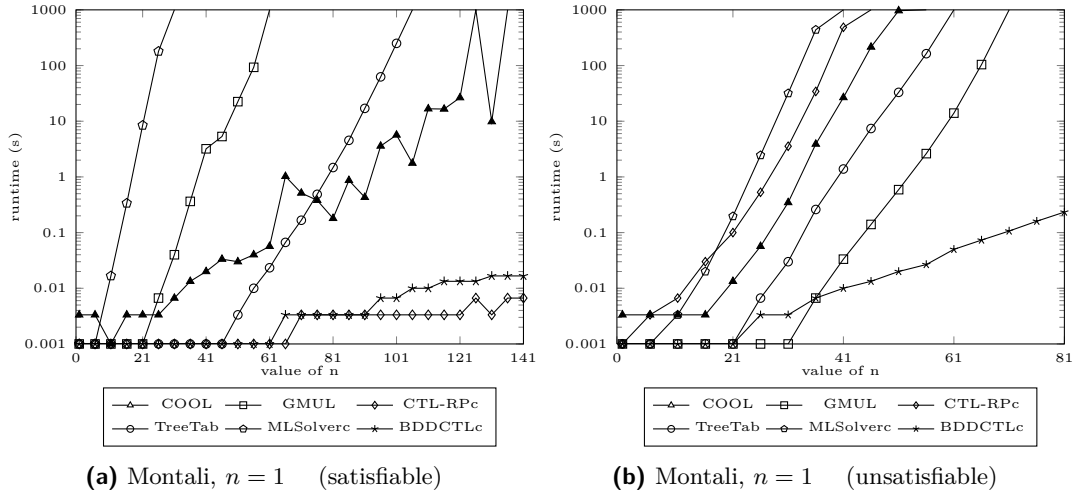


Figure 1 Runtimes for the Montali-formulas

In addition, we design two series of unsatisfiable benchmark formulas that have an exponentially large search space but allow for detection of unsatisfiability at an early stage. Recall that in CTL we can express the statement ‘in the next step, the n -bit counter x represented by the variables x_1, \dots, x_n will be incremented’ (with wraparound) as a formula $c(x, n)$ of polynomial size in n . We define unsatisfiable formulas $early(n, j, k)$ that specify an n -bit

Type of formula	COOL	TreeTab	GMUL	MLSolvenc	BDDCTLc	CTL-RPc
(i)	<0.01	<0.01	<0.01	0.02	<0.01	0.02
(ii)	0.12	–	0.02	0.95	<0.01	0.15
(iii)	0.12	–	0.02	0.87	<0.01	0.16

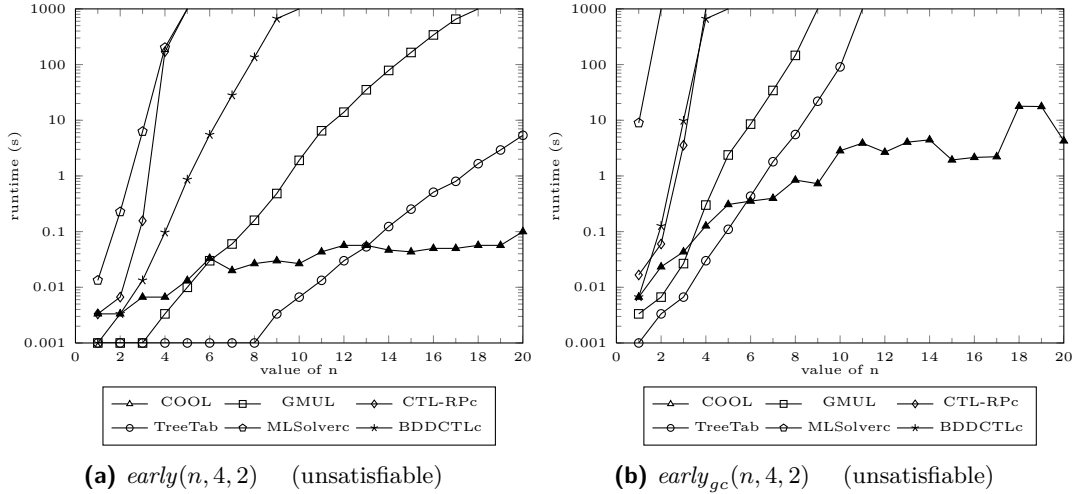
■ **Figure 2** Runtimes (in s) for the Alternating Bit Protocol formulas

counter p with n bits and additionally branch after 2^j steps (i.e. when p_j holds) to start a counter r with k bits which in turn forever postpones the eventuality $EF\ p$:

$$\begin{aligned}
early(n, j, k) &= start_p \wedge init(p, n) \wedge init(r, k) \wedge AG((r \rightarrow c(r, k)) \wedge (p \rightarrow c(p, n))) \wedge \\
&\quad AG((\bigwedge_{0 \leq i \leq j} p_i \rightarrow EX(start_r \wedge EF\ p)) \wedge \neg(p \wedge r) \wedge (r \rightarrow AX\ r)) \\
init(x, m) &= AG((start_x \rightarrow (x \wedge \bigwedge_{0 \leq i < m} \neg x_i)) \wedge (x \rightarrow EX\ x)).
\end{aligned}$$

Note here that $init$ uses x as a string argument; $start_x$ is an atom indicating the start of counter x , and the atom x itself indicates that the counter x is running. The second series of unsatisfiable formulas $early_{gc}(n, j, k)$ is obtained by extending the formulas $early(n, j, k)$ with the additional requirement that a further counter q with n bits is started infinitely often, but at most at every second step:

$$\begin{aligned}
early_{gc}(n, j, k) &= early(n, j, k) \wedge b \wedge init(q, n) \wedge AG(\neg(p \wedge q) \wedge \neg(q \wedge r) \wedge (q \rightarrow c(q, n))) \\
&\quad \wedge AG(AF\ b \wedge (b \rightarrow (EX\ p \wedge EX\ start_q \wedge AX\ \neg b)))
\end{aligned}$$



■ **Figure 3** Formulas with exponential search space and sub-exponential refutations

Figure 3 shows the respective runtimes for these formulas. In all cases, COOL finishes before the tableau is fully expanded, while GMUL and MLSolver will necessarily complete their first pass before being able to decide the formulas, and hence exhibit exponential behaviour; TreeTab seems not to benefit substantially from its capability to close tableaux early. For the $early_{gc}$ formulas, the ability to cache previously seen nodes appears to provide COOL with additional advantages. The $early_{gc}$ series can be converted into satisfiable formulas by replacing AX with EX , with similar results.

Due to the apparent lack of benchmarking formulas for the alternation-free μ -calculus and ATL, we compare runtimes on random formulas for these logics. For the alternation-free μ -calculus, formulas were built from 250 random operators (where disjunction and conjunction

are twice as likely as the other operators). The experiment was conducted with formulas over three and over ten propositional atoms, respectively. MLSolver ran out of memory on 21% on the formulas over three atoms and on 16% of the formulas over ten atoms. COOL answered all queries without exceeding memory restrictions, and in under one second for all queries but one. Altogether, COOL was faster than MLSolver for more than 98% of the random alternation-free formulas, with the median of the ratios of the runtimes being 0.0431 in favour of COOL for formulas over three atoms and 0.0833 for formulas over ten atoms (recall however that MLSolver supports the full μ -calculus). For alternating-time temporal logic ATL, we compared the runtimes of TATL and COOL on random formulas consisting of 50 random operators; COOL answered faster than TATL on all of the formulas, with the median of the ratios of runtimes being 0.000668 in favour of COOL.

6 Conclusion

We have presented a tableau-based global caching algorithm of optimal (EXPTIME) complexity for satisfiability in the alternation-free coalgebraic μ -calculus; the algorithm instantiates to the alternation-free fragments of e.g. the relational μ -calculus, the alternating-time μ -calculus (AMC) and the serial monotone μ -calculus. Essentially, it simultaneously generates and solves a deterministic Büchi game on-the-fly in a direct construction, in particular skipping the determinization of co-Büchi automata; the correctness proof, however, is stand-alone. We have generalized the $2^{\mathcal{O}(n)}$ bound on model size for alternation-free fixpoint formulas from the relational case to the coalgebraic level of generality, in particular to the AMC.

We have implemented the algorithm as part of the generic solver COOL; the implementation shows promising performance for CTL, ATL and the alternation-free relational μ -calculus. An extension of our global caching algorithm to the full μ -calculus would have to integrate Safra-style determinization of Büchi automata [34] and solving of the resulting parity game, both on-the-fly.

References

- 1 Rajeev Alur, Thomas Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49:672–713, 2002.
- 2 Julian Bradfield and Colin Stirling. Modal μ -calculi. In *Handbook of Modal Logic*, pages 721–756. Elsevier, 2006.
- 3 Kai Brännler and Martin Lange. Cut-free sequent systems for temporal logic. *J. Log. Algebr. Prog.*, 76:216–225, 2008.
- 4 Corina Cîrstea, Clemens Kupke, and Dirk Pattinson. EXPTIME tableaux for the coalgebraic μ -calculus. *Log. Meth. Comput. Sci.*, 7, 2011.
- 5 Corina Cîrstea, Alexander Kurz, Dirk Pattinson, Lutz Schröder, and Yde Venema. Modal logics are coalgebraic. *Comput. J.*, 54:31–41, 2011.
- 6 Edmund Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In *Logics of Programs*, volume 131 of *LNCS*, pages 52–71. Springer, 1982.
- 7 Amélie David. TATL: Implementation of ATL tableau-based decision procedure. In *Automated Reasoning with Analytic Tableaux and Related Methods, TABLEAUX 2013*, volume 8123 of *LNCS*, pages 97–103. Springer, 2013.
- 8 E. Allen Emerson and Joseph Halpern. Decision procedures and expressiveness in the temporal logic of branching time. *J. Comput. Sys. Sci.*, 30:1–24, 1985.

- 9 E. Allen Emerson and Charanjit Jutla. The complexity of tree automata and logics of programs. *SIAM J. Comput.*, 29(1):132–158, September 1999.
- 10 Oliver Friedmann and Martin Lange. Local strategy improvement for parity game solving. In *Games, Automata, Logic, and Formal Verification, Gandalf 2010*, volume 25 of *EPTCS*, pages 118–131. Open Publishing Association, 2010.
- 11 Oliver Friedmann and Martin Lange. A solver for modal fixpoint logics. In *Methods for Modalities, M4M-6 2009*, volume 262 of *ENTCS*, pages 99–111, 2010.
- 12 Oliver Friedmann and Martin Lange. Deciding the unguarded modal μ -calculus. *J. Appl. Non-Classical Log.*, 23:353–371, 2013.
- 13 Oliver Friedmann, Markus Latte, and Martin Lange. Satisfiability games for branching-time logics. *Log. Methods Comput. Sci.*, 9, 2013.
- 14 Rajeev Goré. And-Or tableaux for fixpoint logics with converse: LTL, CTL, PDL and CPDL. In *Automated Reasoning, IJCAR 2014*, volume 8562 of *LNCS*, pages 26–45. Springer, 2014.
- 15 Rajeev Goré, Clemens Kupke, and Dirk Pattinson. Optimal tableau algorithms for coalgebraic logics. In *Tools and Algorithms for the Construction and Analysis of Systems, TACAS 2010*, volume 6015 of *LNCS*, pages 114–128. Springer, 2010.
- 16 Rajeev Goré, Clemens Kupke, Dirk Pattinson, and Lutz Schröder. Global caching for coalgebraic description logics. In *Automated Reasoning, IJCAR 2010*, volume 6173 of *LNCS*, pages 46–60. Springer, 2010.
- 17 Rajeev Goré and Linh Anh Nguyen. Exptime tableaux for *ALC* using sound global caching. *J. Autom. Reasoning*, 50:355–381, 2013.
- 18 Rajeev Goré, Jimmy Thomson, and Florian Widmann. An experimental comparison of theorem provers for CTL. In *Temporal Representation and Reasoning, TIME 2011*, pages 49–56. IEEE, 2011.
- 19 Rajeev Goré and Florian Widmann. An optimal on-the-fly tableau-based decision procedure for PDL-satisfiability. In *Automated Deduction, CADE 2009*, volume 5663 of *LNCS*, pages 437–452. Springer, 2009.
- 20 Rajeev Goré and Florian Widmann. Sound global state caching for *ALC* with inverse roles. In *Automated Reasoning with Analytic Tableaux and Related Methods, TABLEUX 2009*, volume 5607 of *LNCS*, pages 205–219. Springer, 2009.
- 21 Daniel Gorín, Dirk Pattinson, Lutz Schröder, Florian Widmann, and Thorsten Wißmann. COOL – a generic reasoner for coalgebraic hybrid logics (system description). In *Automated Reasoning, IJCAR 2014*, volume 8562 of *LNCS*, pages 396–402. Springer, 2014.
- 22 Daniel Hausmann and Lutz Schröder. Global caching for the flat coalgebraic μ -calculus. In *Temporal Representation and Reasoning, TIME 2015*, pages 121–143. IEEE, 2015.
- 23 Natthapong Jungteerapanich. A tableau system for the modal μ -calculus. In *Automated Reasoning with Analytic Tableaux and Related Methods, TABLEUX 2009*, volume 5607 of *LNCS*, pages 220–234. Springer, 2009.
- 24 Dexter Kozen. Results on the propositional μ -calculus. *Theor. Comput. Sci.*, 27:333–354, 1983.
- 25 Dexter Kozen. A finite model theorem for the propositional μ -calculus. *Stud. Log.*, 47:233–241, 1988.
- 26 Martin Lange and Colin Stirling. Focus games for satisfiability and completeness of temporal logic. In *Logic in Computer Science, LICS 2001*, pages 357–365. IEEE Computer Society, 2001.
- 27 Zohar Manna and Amir Pnueli. The modal logic of programs. In *Automata, Languages and Programming, ICALP 1979*, volume 71 of *LNCS*, pages 385–409. Springer, 1979.
- 28 Satoru Miyano and Takeshi Hayashi. Alternating finite automata on ω -words. *Theoret. Comput. Sci.*, 32:321–330, 1984.

- 29 Marco Montali, Paolo Torroni, Marco Alberti, Federico Chesani, Marco Gavanelli, Evelina Lamma, and Paola Mello. Verification from declarative specifications using logic programming. In *Logic Programming, ICLP 2008*, volume 5366 of *LNCS*, pages 440–454. Springer, 2008.
- 30 Damian Niwinski and Igor Walukiewicz. Games for the μ -calculus. *Theor. Comput. Sci.*, 163:99–116, 1996.
- 31 Rohit Parikh. The logic of games and its applications. *Ann. Discr. Math.*, 24:111–140, 1985.
- 32 Amir Pnueli. The temporal logic of programs. In *Foundations of Computer Science, FOCS 1977*, pages 46–57. IEEE Computer Society, 1977.
- 33 Vaughan Pratt. Semantical considerations on Floyd-Hoare logic. In *Foundations of Computer Science, FOCS 1976*, pages 109–121. IEEE Computer Society, 1976.
- 34 Shmuel Safra. On the complexity of omega-automata. In *Foundations of Computer Science, FOCS 1988*, pages 319–327. IEEE Computer Society, 1988.
- 35 Sven Schewe. *Synthesis of distributed systems*. PhD thesis, Universität des Saarlands, 2008.
- 36 Lan Zhang, Ullrich Hustadt, and Clare Dixon. A resolution calculus for the branching-time temporal logic CTL. *ACM Trans. Comput. Log.*, 15, 2014.

A

 Omitted Proofs and Lemmas

A.1 Proofs and Lemmas for Section 2

► **Definition 29.** We let $BV(\psi)$ denote the set of variables X such that ηX occurs in ψ .

► **Lemma 30** (Substitution). *If $BV(\psi) \cap FV(\phi) = \emptyset$, then*

$$\llbracket \psi \rrbracket_i^X \llbracket \phi \rrbracket_i = \llbracket \psi[X \mapsto \phi] \rrbracket_i.$$

Proof. The proof is by induction over ψ . If $\psi = \perp$, $\psi = \top$, $\psi = p$ or $\psi = \neg p$, for $p \in P$, then ψ is closed so that $\llbracket \psi \rrbracket_i^X \llbracket \phi \rrbracket_i = \llbracket \psi \rrbracket_i = \llbracket \psi[X \mapsto \phi] \rrbracket_i$. If $\psi = X$, then $\llbracket X \rrbracket_i^X \llbracket \phi \rrbracket_i = \llbracket \phi \rrbracket_i = \llbracket X[X \mapsto \phi] \rrbracket_i$. If $\psi = Y \neq X$, then $\llbracket Y \rrbracket_i^X \llbracket \phi \rrbracket_i = \llbracket Y \rrbracket_i = \llbracket Y[X \mapsto \phi] \rrbracket_i$. The cases for disjunction, conjunction and modal operators are straightforward. If $\psi = \eta X. \psi_1$, then $\llbracket \eta X. \psi_1 \rrbracket_i^X \llbracket \phi \rrbracket_i = \llbracket \eta X. \psi_1 \rrbracket_i = \llbracket (\eta X. \psi_1)[X \mapsto \phi] \rrbracket_i$. If $\psi = \eta Y. \psi_1$ for $Y \neq X$, then $\llbracket \eta Y. \psi_1 \rrbracket_i^X \llbracket \phi \rrbracket_i = \eta \llbracket \psi_1 \rrbracket_{i[X \mapsto \llbracket \phi \rrbracket_i]}^Y = \eta \llbracket \psi_1[X \mapsto \phi] \rrbracket_i^Y = \llbracket (\eta Y. (\psi_1[X \mapsto \phi])) \rrbracket_i = \llbracket (\eta Y. \psi_1)[X \mapsto \phi] \rrbracket_i$, where the second equality holds since for all A ,

$$\begin{aligned} \llbracket \psi_1 \rrbracket_{i[X \mapsto \llbracket \phi \rrbracket_i]}^Y(A) &= \llbracket \psi_1 \rrbracket_{i[X \mapsto \llbracket \phi \rrbracket_i][Y \mapsto A]} \\ &= \llbracket \psi_1 \rrbracket_{i[Y \mapsto A][X \mapsto \llbracket \phi \rrbracket_i]} \\ &= \llbracket \psi_1 \rrbracket_{i[Y \mapsto A]}^X \llbracket \phi \rrbracket_i \\ &= \llbracket \psi_1 \rrbracket_{i[Y \mapsto A]}^X \llbracket \phi \rrbracket_{i[Y \mapsto A]} \\ &= \llbracket \psi_1[X \mapsto \phi] \rrbracket_{i[Y \mapsto A]} \\ &= \llbracket \psi_1[X \mapsto \phi] \rrbracket_i^Y(A), \end{aligned}$$

where the second equality holds since $X \neq Y$, the fourth equality holds since by assumption, $Y \notin FV(\phi)$ and the fifth equality is by the induction hypothesis. ◀

We note that by Lemma 30,

$$\llbracket \eta X. \psi \rrbracket_i = \eta \llbracket \psi \rrbracket_i^X = \llbracket \psi \rrbracket_i^X \llbracket \eta X. \psi \rrbracket_i = \llbracket \psi[X \mapsto \eta X. \psi] \rrbracket_i.$$

A.2 Proofs and Lemmas for Section 3

In the following we will consider all deferrals to be in *decomposed form*, i.e. given a formula ψ that belongs to some eventuality θ , so that $\psi = \alpha\sigma$ for appropriate α and σ , according to Definition 2, we equivalently represent ψ by the pair (α, σ) . This allows us to directly refer to the *base* α and the *sequence* σ of a deferral. We say that the pair (α, σ) *induces* the formula $\alpha\sigma$.

Proof of Lemma 3: The first part of the Lemma is stated by Lemma 31. The proof of the second part is by lexicographic induction over $(|\sigma|, \alpha)$, distinguishing cases for α . The interesting case is the fixpoint variable case, i.e. $\alpha = Y$ for some Y . If $|\sigma| = 1$, we have that $\sigma = [Y \mapsto \theta]$ and hence $Y\sigma = \theta$. If $|\sigma| > 1$, we have $Y\sigma = \chi\kappa$ where χ is the result of applying the first substitution from σ that touches Y to Y and where κ consists of the remaining substitutions from σ . We have $|\kappa| < |\sigma|$ and (χ, κ) is a θ -deferral so that the induction hypothesis finishes the proof. ◀

► **Lemma 31.** *Let (α, σ) be an θ_1 -deferral and let (β, κ) be an θ_2 -deferral such that $\alpha\sigma = \psi = \beta\kappa$. Then $\theta_1 = \theta_2$.*

Proof. We show that $\theta_2 \leq \theta_1$, the other direction is symmetric. We note that by Lemma 3, $\theta_2 \leq \psi$. If $\theta_2 \leq \alpha$, $\theta_2 < \theta_1$ and hence $\theta_2 \leq \theta_1$, as required. If $\theta_2 \not\leq \alpha$, then let $\theta_2 = \mu Y. \phi$ and $\sigma = [X_1 \mapsto \chi_1]; \dots; [X_n \mapsto \chi_n]$ where $\chi_n = \theta_1$. Since $\theta_2 \leq \psi$ but $\theta_2 \not\leq \alpha$, we are in one of the following two cases: a) There is a variable $X \in FV(\alpha)$ with $\theta_2 \leq X\sigma$ in which case – since θ_2 is irreducible – $\theta_2 \leq \chi_i \leq \theta_1$ for some $1 \leq i \leq n$: otherwise there is some $\chi_j = \mu Y. \phi_1$ such that $\mu Y. \phi_1([X_{j+1} \mapsto \chi_{j+1}]; \dots; [X_n \mapsto \chi_n]) = \theta_2$ which is a contradiction to θ_2 being irreducible; b) The formula α contains a fixpoint literal $\mu Y. \phi_1$ with $\phi_1 \sigma = \phi$. But then $\theta_2 = (\mu Y. \phi_1) \sigma$ and $(\mu Y. \phi_1, \sigma)$ is a sequence over χ_n which is a contradiction to θ_2 being irreducible. \blacktriangleleft

Proof of Lemma 10: Note that

$$\begin{aligned} \hat{f}_{X'}(Y') &= (f(X' \cap Y') \cap \overline{F}) \cup (f(X') \cap F) \\ &\subseteq (f(X \cap Y) \cap \overline{F}) \cup (f(X) \cap F) \\ &= \hat{f}_X(Y) \end{aligned}$$

where the inclusion holds since $X' \cap Y' \subseteq X \cap Y$ and since f is monotone w.r.t. set inclusion so that $f(X' \cap Y') \subseteq f(X \cap Y)$ and $f(X') \subseteq f(X)$. The proof for \hat{g} is analogous. \blacktriangleleft

Proof of Lemma 13: Let $G' \subseteq G$. We show $E_{G'} \subseteq E_G$, the proof of $A_{G'} \subseteq A_G$ is analogous. We denote by f_C , and $(\hat{f}_X)_C$ the respective transitionals with base set $C \subseteq G$ and note that for all $X, Y \subseteq G$,

$$f_{G'}(Y) \subseteq f_G(Y) \quad \text{and} \quad (\hat{f}_X)_{G'}(Y) \subseteq (\hat{f}_X)_G(Y).$$

From this we obtain $\mu((\hat{f}_X)_{G'}) \subseteq \mu((\hat{f}_X)_G)$ by induction; this in turn implies that for all Y , $(X \mapsto \mu((\hat{f}_X)_{G'}))Y \subseteq (X \mapsto \mu((\hat{f}_X)_G))Y$. Induction yields $\nu(X \mapsto \mu((\hat{f}_X)_{G'})) \subseteq \nu(X \mapsto \mu((\hat{f}_X)_G))$, as required. \blacktriangleleft

► **Lemma 32.** Let $G \subseteq \mathbf{N}$ be fully expanded and let $C \subseteq \mathbf{C}_G$ be the base set of f and g . For all sets $Y \subseteq C$,

$$f(Y) = \overline{g(\overline{Y})},$$

where for each $Y' \subseteq C$, $\overline{Y'}$ denotes the complement of Y' in C .

Proof. The inclusion “ \subseteq ” is immediate. For the inclusion “ \supseteq ”, let $(\Delta, d) \in \overline{g(\overline{Y})}$ so that it is not the case that there is a $\Sigma \in Cn(\Delta)$ such that for each $\Gamma \in \Sigma$, $(\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in \overline{Y}$. Since G is fully expanded, this implies that for all $\Sigma \in Cn(\Delta)$, there is a $\Gamma \in \Sigma$ such that $(\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in Y$, i.e. that $(\Delta, d) \in f(Y)$. \blacktriangleleft

► **Lemma 33.** If $G \subseteq \mathbf{N}$ is fully expanded and $C \subseteq \mathbf{C}_G$ is the base set of \hat{f}_X and $\hat{g}_{\overline{X}}$, then for all sets of nodes $Y \subseteq C$,

$$\hat{f}_X(Y) = \overline{\hat{g}_{\overline{X}}(\overline{Y})}.$$

Proof. Just note that

$$\begin{aligned} \hat{f}_X(Y) &= (f(X \cap Y) \cap \overline{F}) \cup (f(X) \cap F) \\ &= \overline{(g(\overline{X \cap Y}) \cup F) \cap (g(\overline{X}) \cup \overline{F})} \\ &= \overline{\hat{g}_{\overline{X}}(\overline{Y})}. \end{aligned}$$

where the second equality follows, as G is fully expanded, from Lemma 32. \blacktriangleleft

Proof of Lemma 14: We obtain $E_G = \nu(X \mapsto \mu(\hat{f}_X)) = \overline{\mu(X \mapsto \nu(\hat{g}_X))} = \overline{A_G}$ from Lemma 33 which states that $\hat{f}_X(Y) = \hat{g}_X(\overline{Y})$ for all $X \subseteq \mathbf{C}_G$ in combination with the fact that for complementary monotone functions f and g , $\mu f = \overline{\nu g}$. ◀

Proof of Lemma 15: Let G denote the set of nodes which is created by the algorithm without intermediate propagation – i.e. without step 3) – and notice that G is fully expanded. Let $(\{\phi_0\}, d(\{\phi_0\})) \in E_G$ and let G_p be the set of nodes created by any run of the algorithm (possibly involving intermediate propagation). We note that $G_p \subseteq G$ so that Lemma 13 tells us that $A_{G_p} \subseteq A_G$. As G is fully expanded, Lemma 14 states that $A_G = \overline{E_G}$. As $(\{\phi_0\}, d(\{\phi_0\})) \in E_G$, $(\{\phi_0\}, d(\{\phi_0\})) \notin A_{G_p} \subseteq A_G = \overline{E_G}$, as required. ◀

A.3 Proofs and Lemmas for Section 4

Throughout this subsection, we fix $N \subseteq \mathbf{N}$ to be the fully expanded set of nodes constructed by a run of the algorithm without intermediate propagation.

► **Definition 34.** Given a substitution σ , we define the *domain* $\text{dom}(\sigma)$ of σ as the set of all fixpoint variables that σ *touches*, i.e. the set of all fixpoint variables X with $\sigma(X) \neq X$.

Regarding Definition 21, we note that for all $\Gamma \in N$, all eventualities θ and all deferrals δ , since $d(\Gamma) \cup N(\Gamma, \theta) = \Gamma$, we have $d(\Gamma) \vdash_{\Gamma} \delta$ iff $\Gamma \vdash_{PL} \delta$.

► **Lemma 35** (Syntactic substitution). *If $(\{X\} \cup BV(\psi)) \cap \text{dom}(\sigma) = \emptyset$ and for each $Y \in FV(\psi)$, $(\{X\} \cup BV(\psi)) \cap FV(\sigma(Y)) = \emptyset$,*

$$(\psi\sigma)[X \mapsto (\phi\sigma)] = (\psi[X \mapsto \phi])\sigma.$$

Proof. The proof is by induction over ψ . If $\psi = \perp$, $\psi = \top$, $\psi = p$ or $\psi = \neg p$, for $p \in P$, then ψ is closed and hence $(\psi\sigma)[X \mapsto (\phi\sigma)] = \psi = (\psi[X \mapsto \phi])\sigma$. If $\psi = X$, then note that by assumption $X \notin \text{dom}(\sigma)$ so that $(X\sigma)[X \mapsto (\phi\sigma)] = X[X \mapsto \phi\sigma] = \phi\sigma = (X[X \mapsto \phi])\sigma$. If $\psi = Y \neq X$, then we have by assumption $X \notin FV(\sigma(Y))$ so that $(Y\sigma)[X \mapsto (\phi\sigma)] = \sigma(Y)[X \mapsto \phi\sigma] = \sigma(Y) = Y\sigma = (Y[X \mapsto \phi])\sigma$. The cases for conjunction, disjunction and modal operators are straightforward. If $\psi = \eta X. \psi$, then $((\eta X. \psi)\sigma)[X \mapsto (\phi\sigma)] = (\eta X. \psi)\sigma = ((\eta X. \psi)[X \mapsto \phi])\sigma$. If $\psi = \eta Y. \psi$ for $X \neq Y$, then we have by assumption that $Y \notin \text{dom}(\sigma)$ and for any $Z \in FV(\psi)$, $Y \notin FV(\sigma(Z))$ so that $((\eta Y. \psi)\sigma)[X \mapsto (\phi\sigma)] = \eta Y. (\psi\sigma)[X \mapsto (\phi\sigma)] = \eta Y. (\psi\sigma[X \mapsto (\phi\sigma)]) = \eta Y. ((\psi[X \mapsto \phi])\sigma) = (\eta Y. (\psi[X \mapsto \phi]))\sigma = ((\eta Y. \psi)[X \mapsto \phi])\sigma$, where the third equality is by the induction hypothesis. ◀

► **Definition 36.** Let t_1 and t_2 be unfolding trees for ψ and ϕ . Define $t_1[X \mapsto t_2]$ as the unfolding tree for $\psi[X \mapsto \phi]$ that is obtained by replacing every node in t_1 that represents a free occurrence of X in ψ with t_2 .

► **Lemma 37.** *For each state x and each formula ψ such that $x \models \psi$, there is a least unfolding tree t such that $x \models \psi(t)$.*

Proof. We construct t by walking from left to right through all paths in the syntax tree of ψ , assigning numbers to nodes that represent least fixpoint literals. Let κ be a position and let t_κ denote the tree that has been constructed so far on the walk from the root of the syntax tree to κ . We assign n_κ to the node at position κ if that node represents a least fixpoint literal $\mu X_\kappa. \psi_\kappa$ where n_κ is the least number such that $x \models c_\kappa((\psi_\kappa)_{X_\kappa}^{n_\kappa})$, where

$\psi = c(\mu X_\kappa. \psi_\kappa)$ and where c_κ denotes the context that is obtained from c by replacing any least fixpoint literal $\mu X_\rho. \psi_\rho \leq c$ that already has a number n_ρ assigned to it in t_κ by $(\psi_\rho)_{X_\rho}^{n_\rho}$ and by replacing any other fixpoint literals in c by their n -th unfolding, where n is the size of the finite model. The unfolding tree that we obtain is by construction the least (w.r.t $<_\psi$) unfolding tree t for ψ such that $x \models \psi(t)$. \blacktriangleleft

► **Lemma 38.** *For all n , if $X \neq Y$,*

$$(\psi[X \mapsto \phi])_Y^n = \psi_Y^n[X \mapsto \phi].$$

Proof. By induction over n . If $n = 0$, $\perp = \perp$. Otherwise

$$\begin{aligned} (\psi[X \mapsto \phi])_Y^n &= (\psi[X \mapsto \phi])_Y((\psi[X \mapsto \phi])_Y^{n-1}) \\ &= (\psi[X \mapsto \phi])_Y(\psi_Y^{n-1}[X \mapsto \phi]) \\ &= (\psi_Y(\psi_Y^{n-1}))[X \mapsto \phi] = \psi_Y^n[X \mapsto \phi], \end{aligned}$$

where the second equality is by the induction hypothesis and the third equality is by Lemma 35. \blacktriangleleft

► **Lemma 39.** *Let t_1 be an unfolding tree for ψ and let t_2 be an unfolding tree for ϕ . Then*

$$(\psi[X \mapsto \phi])(t_1[X \mapsto t_2]) = (\psi(t_1))[X \mapsto \phi(t_2)].$$

Proof. The proof is by standard induction over ψ . We consider the only interesting case, i.e. the case that $\psi = \mu Y. \psi_1$ where $X \neq Y$. Then

$$\begin{aligned} (\mu Y. \psi_1[X \mapsto \phi])(t_1[X \mapsto t_2]) &= (\mu Y. (\psi_1[X \mapsto \phi]))(t_1[X \mapsto t_2]) \\ &= ((\psi_1[X \mapsto \phi])(t_3[X \mapsto t_2]))_Y^n \\ &= ((\psi_1(t_3))[X \mapsto \phi(t_2)])_Y^n \\ &= ((\psi_1(t_3)))_Y^n[X \mapsto \phi(t_2)] \\ &= (\mu Y. \psi_1(t_3))[X \mapsto \phi(t_2)] \end{aligned}$$

where t_3 is the child of the root of t_1 . The third equality is by the induction hypothesis and the fourth equality is by Lemma 38. \blacktriangleleft

► **Lemma 40.** *Let t and s be unfolding trees for $\phi_1 = \eta X. \psi \sigma$ and $\phi_2 = \psi(\eta X. \psi, \sigma)$, respectively. Furthermore, let $t(\epsilon, \phi_1) = n + 1$ and $s(\tau, \phi_1) = n$ for all positions τ at which ϕ_1 occurs in ϕ_2 ; also let $t(\kappa, \chi) = s(\tau, \chi)$ for all least fixpoint literals χ occurring in ϕ_1 at some position $\kappa \neq \epsilon$ and all τ such that χ occurs in ϕ_2 at position τ and either $\kappa = 0\tau$ or $\tau = \rho\kappa$ where X occurs freely in ψ at position ρ . Then*

$$x \models \eta X. \psi \sigma(t) \text{ implies } x \models (\psi(\eta X. \psi, \sigma))(s).$$

Proof. So let $t(\epsilon, \eta X. \psi \sigma) = n + 1 = s(\tau, \eta X. \psi \sigma) + 1$ for all appropriate τ . Let t_1 denote the child of the root of t and let s_1, s_2 and s_3 denote subtrees of s such that $s = s_1[X \mapsto s_2]$ and s_3 is the child of the root of s_2 . Then

$$\begin{aligned} \eta X. \psi \sigma(t) &= (\psi \sigma(t_1))_X^{n+1} \\ &= (\psi \sigma(t_1))_X((\psi \sigma(t_1))_X^n) \end{aligned}$$

and

$$\begin{aligned}
(\psi(\eta X. \psi, \sigma))(s) &= ((\psi[X \mapsto \eta X. \psi])\sigma)(s) \\
&= (\psi\sigma[X \mapsto \eta X. \psi\sigma])(s) \\
&= (\psi\sigma(s_1))([X \mapsto \eta X. \psi\sigma](s_2)) \\
&= (\psi\sigma(s_1))_X(\eta X. \psi\sigma(s_2)) \\
&= (\psi\sigma(s_1))_X((\psi\sigma(s_3))_X^n),
\end{aligned}$$

where the fifth equality holds since $s_2(\epsilon, \eta X. \psi\sigma) = n$. As $\psi\sigma$ does not contain $\eta X. \psi\sigma$ and s and t agree on all other fixpoint literals, $t_1 = s_1 = s_3$, which finishes the proof. \blacktriangleleft

► **Definition 41** (Realization). The set of \mathcal{K} -realized nodes is

$$M = \{(\Gamma, d) \mid \Gamma \in N, d \subseteq d(\Gamma), \exists x \in W. \forall \phi. \Gamma \vdash_{PL} \phi \Rightarrow x \models_W \phi\}.$$

► **Definition 42** (Rank). The *rank* $\text{rk}(\psi)$ of a formula ψ is the depth of nesting of modal operators in it. Given a set d of deferrals and a state $x \in W$ such that $x \models \alpha\sigma$ for each $(\alpha, \sigma) \in d$, we put

$$\text{rk}(d, x) = \max\{\text{rk}(\text{unf}((\alpha, \sigma), x)) \mid (\alpha, \sigma) \in d\}.$$

For $(\Gamma, d') \in M$, we put

$$\text{rk}(d, \Gamma) = \min\{\text{rk}(d, x) \mid \forall \phi. \Gamma \vdash_{PL} \phi \Rightarrow x \models \phi\}.$$

► **Corollary 43.** Let $x \models (\eta X. \psi)\sigma$. Then

$$\text{rk}(\text{unf}((X, (\eta X. \psi, \sigma), x))) \geq \text{rk}(\text{unf}((\psi, (\eta X. \psi, \sigma), x))).$$

Proof. Let t and s be the least unfolding trees for $X(\eta X. \psi, \sigma) = \eta X. \psi\sigma$ and $\psi(\eta X. \psi, \sigma)$ such that $x \models \eta X. \psi\sigma(t)$ and $x \models (\psi(\eta X. \psi, \sigma))(s)$, respectively. Lemma 40 finishes the proof as it states that s can be chosen to agree with t on all least fixpoint literals except for $\eta X. \psi\sigma$ for which we have $t(\epsilon, \eta X. \psi\sigma) = s(\kappa, \eta X. \psi\sigma) + 1$ for any suitable κ ; thus $(\psi(\eta X. \psi, \sigma))(s)$ has a rank that is not greater than the rank of $\eta X. \psi\sigma(t)$, as required. \blacktriangleleft

► **Lemma 44.** For all deferrals (α, σ) and all unfolding trees $t_{\alpha\sigma}$,

$$\llbracket \alpha\sigma(t_{\alpha\sigma}) \rrbracket \subseteq \llbracket \alpha\sigma \rrbracket.$$

Proof. This lemma follows by induction over $\alpha\sigma$ from $\llbracket \psi_X^n \rrbracket \subseteq \llbracket \mu X. \psi \rrbracket$. \blacktriangleleft

► **Definition 45** ((Pseudo-)Theory). We define the *pseudo-theory* $\Gamma \vdash_{PL}$ of a node $\Gamma \in N$ as

$$\Gamma \vdash_{PL} = \{\phi \in \mathbf{F} \mid \Gamma \vdash_{PL} \phi\},$$

and the *theory* $x \models$ of a state $x \in W$ as

$$x \models = \{\phi \in \mathbf{F} \mid x \models \phi\}.$$

Given a node $\Gamma \in N$ and a state $x \in W$, we write $\Gamma \subseteq x$ if $(\Gamma \vdash_{PL}) \subseteq (x \models)$, equivalently $\Gamma \subseteq (x \models)$.

Recall that M denotes the set of \mathcal{K} -realized nodes (cf. Definition 41) and note that

$$M = \{(\Gamma, d) \mid \Gamma \in N, d \subseteq d(\Gamma), \exists x \in W. \Gamma \subseteq x\}.$$

► **Lemma 46.** *Let $x \in W$, $(\Delta, d) \in M \cap \mathbf{S} \times \mathbf{S}$ and $\Delta \subseteq x$. Given a set $B_{\langle a \rangle \alpha} \subseteq W$ for each $\langle a \rangle \alpha \in \Delta$, a set $B_{[a] \alpha} \subseteq W$ for each $[a] \alpha \in \Delta$ such that*

$$\begin{aligned} \langle a \rangle \alpha \in \Delta &\Rightarrow \exists y \in R_a(x). y \in B_{\langle a \rangle \alpha} \\ [a] \alpha \in \Delta &\Rightarrow \forall y \in R_a(x). y \in B_{[a] \alpha}, \end{aligned}$$

and a modal rule

$$(\Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \psi / \psi_1, \dots, \psi_n, \psi)$$

with $\Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \psi = \Delta$, we have $\{\psi_1, \dots, \psi_n, \psi\} = \Theta \in N$ and there is a state $z \in W$ such that $\Theta \subseteq z$ and $z \in \bigcap_{1 \leq i \leq n} B_{[a]\psi_i} \cap B_{\langle a \rangle \psi}$.

Proof. As N is fully expanded, $\{\psi_1, \dots, \psi_n, \psi\} = \Theta \in N$. As $\langle a \rangle \psi \in \Delta$, there is by assumption a state $z \in B_{\langle a \rangle \psi}$. Since $[a]\psi_i \in \Delta$ for $1 \leq i \leq n$, we have by assumption that z is also contained in $\bigcap_{1 \leq i \leq n} B_{[a]\psi_i}$, as required. ◀

► **Definition 47.** We denote by $u_f(\phi)$ and $u_p(\phi)$ the numbers of unguarded occurrences of fixpoint and propositional operators in ϕ , respectively.

Proof of Theorem 19: It suffices to show that \mathcal{K} -realized nodes are successful, i.e. $M \subseteq E_S = \nu(X \mapsto \mu(\hat{f}_X))$. We use coinduction, i.e. show that M is a postfixpoint of $(X \mapsto \mu(\hat{f}_X))$, i.e. $(\Delta, d) \in \mu(\hat{f}_M)$ for all $(\Delta, d(\Delta)) \in M$. We show the more general property that for all $\Delta \in N$ and all $d \subseteq d(\Delta)$, $(\Delta, d) \in \mu(\hat{f}_M)$ and proceed by induction over the triple $(\text{rk}(d, \Delta), u_f(\Delta), u_p(\Delta))$ in lexicographic order $<_l$. If $d = \emptyset$, then $(\Delta, d) \in \hat{f}_M(\mu(\hat{f}_M))$ if $(\Delta, d) \in f(M)$ which is implied by Lemma 48. If $d \neq \emptyset$, $\text{rk}(d, \Delta) > 0$. We distinguish two cases:

- If Δ is a not state node, then let y be a state with $\Delta \subseteq y$. We note that $u_f(\Delta) > 0$ or $u_p(\Delta) > 0$. Let $\Delta = \{\phi_1, \dots, \phi_o\}$. In order to show that $(\Delta, d) \in \hat{f}_M(\mu(\hat{f}_M))$, we consider any non-modal rule that matches Δ and show that it has a conclusion Θ such that $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in \mu(\hat{f}_M)$. To this end we distinguish upon the rule that is being applied.
 - (\perp) , $(p, \neg p)$: These rules are not applicable to Δ since $\Delta \subseteq y$ and $y \not\models \perp$ as well as $y \not\models p \wedge \neg p$ for any p .
 - (\wedge) : Then there is a formula $\phi_i = \psi_1 \wedge \psi_2 \in \Delta$ and the rule leads – since N is fully expanded – to the node $\Theta \in N$ with

$$\Theta = \{\phi_1, \dots, \phi_{i-1}, \psi_1, \psi_2, \phi_{i+1}, \dots, \phi_o\}.$$

We note that $u_f(\Theta) = u_f(\Delta)$, $u_p(\Theta) < u_p(\Delta)$ and $\Theta \subseteq y$, i.e. $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in M$; also $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, \Theta) \leq \text{rk}(d, \Delta)$. By the induction hypothesis, $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in \mu(\hat{f}_M)$, as required.

- (\vee) : Then there is a formula $\phi_i = \psi_1 \vee \psi_2 \in s$ and the rule leads – since N is fully expanded – to the two nodes $\Theta_1, \Theta_2 \in N$ with

$$\Theta_1 = \{\phi_1, \dots, \phi_{i-1}, \psi_1, \phi_{i+1}, \dots, \phi_o\} \quad \text{and}$$

$$\Theta_2 = \{\phi_1, \dots, \phi_{i-1}, \psi_2, \phi_{i+1}, \dots, \phi_o\}.$$

We note that $u_f(\Theta_1) = u_f(\Theta_1) = u_f(\Delta)$, $u_p(\Theta_1) < u_p(\Delta)$ and $u_p(\Theta_2) < u_p(\Delta)$; also $\Theta_1 \subseteq y \models$ or $\Theta_2 \subseteq y \models$ so that there is an $i \in \{1, 2\}$ with $\Theta_i \subseteq y$, i.e. with $(\Theta_i, d_{\Delta \rightsquigarrow \Theta_i}) \in M$; furthermore, $\text{rk}(d_{\Delta \rightsquigarrow \Theta_i}, \Theta_i) \leq \text{rk}(d, \Delta)$. By the induction hypothesis, $(\Theta_i, d_{\Delta \rightsquigarrow \Theta_i}) \in \mu(\hat{f}_M)$, as required.

- (η) : Then there is a formula $\phi_i = \eta X.\psi \in \Delta$ and the rule leads – since N is fully expanded – to the node $\Theta \in N$ with

$$\Theta = \{\phi_1, \dots, \phi_{i-1}, \psi[X \mapsto \eta X.\psi], \phi_{i+1}, \dots, \phi_o\}.$$

We note that $u_f(\Theta) < u_f(\Delta)$ and $\Theta \subseteq y$ so that $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in M$. Let χ abbreviate $\eta X.\psi$; if $\eta = \nu$, χ is not induced by any deferral from d so that $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, \Theta) = \text{rk}(d, \Delta)$. If $\eta = \mu$, then we show that $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, \Theta) \leq \text{rk}(d, \Delta)$. Notice that we can choose a sequence $\sigma = [X_1 \mapsto \chi_1]; \dots; [X_n \mapsto \chi_n]$ that sequentially unfolds some eventuality χ_n and a formula ψ_1 such that $\mu X.\psi_1 <_f \chi_1$ and $\psi_1 \sigma = \psi$; then $(X, [X \mapsto \mu X.\psi_1]; \sigma)$ is a deferral that induces $\chi = \mu X.\psi_1 \sigma$ and $(\psi_1, [X \mapsto \mu X.\psi_1]; \sigma)$ is a deferral that induces $(\psi_1 [X \mapsto \mu X.\psi_1]) \sigma = \psi [X \mapsto \mu X.\psi]$ so that if $(X, [X \mapsto \mu X.\psi_1]; \sigma) \in d$, $(\psi_1, [X \mapsto \mu X.\psi_1]; \sigma) \in d_{\Delta \rightsquigarrow \Theta}$. By Corollary 43, $\text{rk}(\text{unf}((X, [X \mapsto \mu X.\psi_1]; \sigma), y)) \geq \text{rk}(\text{unf}((\psi_1, [X \mapsto \mu X.\psi_1]; \sigma), y))$ which implies – since $(X, [X \mapsto \mu X.\psi_1]; \sigma)$ is the only deferral that changed from Δ to Θ – that we have $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, \Theta) \leq \text{rk}(d, \Delta)$. The induction hypothesis implies $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in \mu(\hat{f}_M)$, as required.

- If Δ is a state node, then let x be a state with $\Delta \subseteq x$ and $\text{rk}(d, \Delta) = \text{rk}(d, x)$. In order to show that $(\Delta, d) \in \hat{f}_M(\mu(\hat{f}_M))$, we show that for all modal rules that match Δ , there is a conclusion Θ of the rule application with $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in \mu(\hat{f}_M)$. Consider any $(\langle a \rangle)$ -rule

$$(\Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \psi / \psi_1, \dots, \psi_n, \psi)$$

with $\Delta = \Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \psi$. We define for each $(\langle a \rangle \beta, \sigma) \in d$ the set $B_{\langle a \rangle \beta \sigma} = \llbracket \beta \sigma(t) \rrbracket$ where $\text{unf}((\langle a \rangle \beta, \sigma), x) = \langle a \rangle \beta \sigma(t)$. We also define for each $([a]\beta, \sigma) \in d$ the set $B_{[a]\beta \sigma} = \llbracket \beta \sigma(t) \rrbracket$ where $\text{unf}([a]\beta, \sigma), x) = [a]\beta \sigma(t)$. By Fact 44, $\llbracket \beta \sigma(t) \rrbracket \subseteq \llbracket \beta \sigma \rrbracket$. For each $\langle a \rangle \beta \in \Delta$ that is not induced by a deferral from d , we define $B_{\langle a \rangle \beta} = \llbracket \beta \rrbracket$, and analogously we put $B_{[a]\beta} = \llbracket \beta \rrbracket$ for each $[a]\beta \in \Delta$ that is not induced by a deferral from d . Note how for each $\langle a \rangle \beta \in \Delta$, there is an $y \in R_a(x)$ with $y \in B_{\langle a \rangle \beta}$: If $\langle a \rangle \beta \in \Delta$ is not induced by a deferral, note that $\Delta \subseteq x$ so that $x \in \llbracket \langle a \rangle \beta \rrbracket$. Otherwise, note that $B_{\langle a \rangle \beta \sigma} = \llbracket \beta \sigma(t) \rrbracket$ where $x \in \llbracket \langle a \rangle \beta \sigma(t) \rrbracket$ which is the case iff there is a $y \in R_a(x)$ with $y \in \llbracket \beta \sigma(t) \rrbracket = B_{\langle a \rangle \beta \sigma}$, as required. For each $[a]\beta \in s$, one shows analogously that for all $y \in R_a(x)$, $y \in B_{[a]\beta}$. Thus by Lemma 46, $\{\psi_1, \dots, \psi_n, \psi\} = \Theta \in M$ and there is a state $z \in W$ with $\Theta \subseteq z$ such that $\bigcap_{1 \leq i \leq n} B_{[a]\psi_i} \cap B_{\langle a \rangle \psi}$. The induction hypothesis implies $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in \mu(\hat{f}_M)$ if $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, \Theta) < \text{rk}(d, \Delta)$. We convince ourselves that indeed $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, \Theta) \leq \text{rk}(d_{\Delta \rightsquigarrow \Theta}, y) < \text{rk}(d, x) = \text{rk}(d, \Delta)$: Recall that $\text{rk}(d_{\Delta \rightsquigarrow \Theta}, y) = \max\{\text{rk}(\text{unf}((\alpha, \sigma), y)) \mid (\alpha, \sigma) \in d_{\Delta \rightsquigarrow \Theta}\}$. Take any $(\alpha, \sigma) \in d_{\Delta \rightsquigarrow \Theta}$ for which $\text{rk}(\text{unf}((\alpha, \sigma), y)) = \text{rk}(d_{\Delta \rightsquigarrow \Theta}, y)$ and consider $(\langle a \rangle \alpha, \sigma) \in d$ (the case for $([a]\alpha, \sigma) \in d$ is analogous, using the upcoming argumentation); if no such deferral exists, $d_{\Delta \rightsquigarrow \Theta} = \emptyset$ and Lemma 48 finishes the proof. Otherwise let $p = \text{rk}(\text{unf}((\langle a \rangle \alpha, \sigma), x))$ and let $q = \text{rk}(\text{unf}((\alpha, \sigma), y))$. Recall that $y \in B_{\langle a \rangle \alpha \sigma} = \llbracket \alpha \sigma(t) \rrbracket$ so that $\text{rk}(\text{unf}((\alpha, \sigma), y)) \leq \text{rk}(\alpha \sigma(t))$ and hence $q < p$. Thus $\text{rk}(\text{unf}((\alpha, \sigma), y)) < \text{rk}(\text{unf}((\langle a \rangle \alpha \sigma), x))$. Hence

$$\begin{aligned} \text{rk}(d_{\Delta \rightsquigarrow \Theta}, y) &= \text{rk}(\text{unf}((\alpha, \sigma), y)) \\ &< \text{rk}(\text{unf}((\langle a \rangle \alpha, \sigma), x)) \\ &\leq \text{rk}(d, x), \end{aligned}$$

as required.

This finishes the proof. \blacktriangleleft

► **Lemma 48.** *For each focused node $(\Delta, d) \in M$ and each $\Sigma \in Cn(\Delta)$, there is a $\Theta \in \Sigma$ such that $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in M$.*

Proof. Let $(\Delta, d) \in M$ and $\Sigma \in Cn(\Delta)$. If Δ is a state node, Σ contains just the conclusion Θ of a modal rule $(\Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \psi / \psi_1, \dots, \psi_n, \psi := \Theta)$ with $\Delta = \Gamma, [a]\psi_1, \dots, [a]\psi_n, \langle a \rangle \psi$. Since N is fully expanded, $\Theta \in N$. As $(\Delta, d) \in M$, there is a state x such that $x \models \langle a \rangle \psi$, i.e. there is a state $y \in R_a(x)$ such that $y \models \psi$. As $x \models [a]\psi_i$, $y \models \psi_i$, for $1 \leq i \leq n$, so that $\Theta \subseteq x$, showing $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in M$, as required. If Δ is not a state node, just note that for all y , $y \models$ is closed under propositional breakdown and unfolding of fixpoint literals. \blacktriangleleft

► **Definition 49.** A finite set of formulas Ψ *propositionally entails* a finite set Φ of formulas (written $\Psi \vdash_{PL} \Phi$) if $\Psi \vdash_{PL} \bigwedge \Phi$.

Proof of Lemma 23: Recall that $E = E_G$. First note that $|D| \leq |E| \leq 3^{|\phi_0|}$. We proceed in two steps: in the first step, we construct a relation $L \subseteq D \times D$; in the second step, we show that L is a timed-out tableau.

1. For any $(\Delta, d) \in D$, $(\Delta, d) \in E = \nu(X \mapsto \mu(\hat{f}_X)) = (X \mapsto \mu(\hat{f}_X))(E) = \mu(\hat{f}_E) = (\hat{f}_E)^n(\emptyset)$ for some n . Let $Cn(\Delta) = \{\Sigma_1, \dots, \Sigma_j\}$. If $n = 0$, $(\Delta, d) \notin (\hat{f}_E)^0(\emptyset) = \emptyset$ so that there is nothing to show. If $n > 0$, $(\Delta, d) \in \hat{f}_E((\hat{f}_E)^{n-1}(\emptyset))$. If $d = \emptyset$, then $(\Delta, d) \in f(E) \cap F$, i.e. there is, for each i , a $\Gamma \in \Sigma_i$ such that $(\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in E$. Notice that since $d = \emptyset$, $d_{\Delta \rightsquigarrow \Gamma} = d(\Gamma)$. As $(\Delta, d) \in (\hat{f}_E)^n(\emptyset)$, this implies by Lemma 50 that there is a state node Θ_i with $\Theta_i \vdash_{PL} \Gamma$. Notice that $d(\Theta_i) \vdash_{\Theta_i} d(\Gamma)$. Put $L(\Delta, d) = \{(\Theta_1, d(\Theta_1)), \dots, (\Theta_j, d(\Theta_j))\}$. If $d \neq \emptyset$, $(\Delta, d) \in f((\hat{f}_E)^{n-1}(\emptyset))$, i.e. there is, for each i , a $\Gamma \in \Sigma_i$ such that $(\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in (\hat{f}_E)^{n-1}(\emptyset)$. If $n - 1 = 0$, $Cn(\Delta) = \emptyset$ and we put $L(\Delta, d) = \emptyset$. Otherwise Lemma 50 implies that there is a state node Θ_i with $\Theta_i \vdash_{PL} \Gamma$ and a set $d_i \subseteq d(\Theta_i)$ with $d_i \vdash_{\Theta_i} d_{\Delta \rightsquigarrow \Gamma}$; for step 2), we note that the Lemma also tells us that $(\Theta_i, d_i) \in (\hat{f}_E)^{n-1}(\emptyset)$. Put $L(\Delta, d) = \{(\Theta_1, d_1), \dots, (\Theta_j, d_j)\}$.
2. We show that L is a timed-out tableau by proving the stronger property that for all $(\Delta, d) \in D$ and all $d' \subseteq d(\Delta)$, there is some m such that $(\Delta, d) \in to(d', m)$. To this end we distinguish two cases. In case a), $d = d'$, while in case b), $d \neq d'$. In both cases, $(\Delta, d) \in E = \nu(X \mapsto \mu(\hat{f}_X)) = (X \mapsto \mu(\hat{f}_X))(E) = \mu(\hat{f}_E) = (\hat{f}_E)^n(\emptyset)$ for some n . If $d' = \emptyset$, $(\Delta, d) \in to(\emptyset, m) = D$ for any m and we are done. If $d' \neq \emptyset$, then we proceed by induction over n . Let $L(\Delta, d) = \{(\Theta_1, d_1), \dots, (\Theta_j, d_j)\}$. If $n = 0$, $Cn(\Delta) = L(\Delta, d) = \emptyset$ in which case there is nothing to show, or $(\Delta, d) \in f(E) \cap F$, so that $d = \emptyset$. Considering the latter situation, if we are in case a), $d' = \emptyset$ and $(\Delta, \emptyset) \in to(\emptyset, m) = D$ for any m so that we are done. If we are in case b), recall from step 1) that $d_1 = d(\Theta_1), \dots, d_j = d(\Theta_j)$; we proceed as in case a), having to show that for all $1 \leq i \leq j$, $(\Theta_i, d_i) \in to(d_i, m)$ for some m . If $n > 0$, recall from step 1) that $L(\Delta, d) = \{(\Theta_1, d_1), \dots, (\Theta_j, d_j)\}$, where $(\Theta_i, d_i) \in (\hat{f}_E)^{n-1}(\emptyset)$. By the induction hypothesis, $(\Theta_i, d_i) \in to(d(\Theta_i), m)$ for some m , as required.

Thus we have constructed a relation L over D – where D has size at most $3^{|\phi_0|}$ – and shown it to be a timed-out tableau. \blacktriangleleft

► **Lemma 50.** *Given a set $X \subseteq \mathbf{C}_G$ and a focused node $(\Delta, d) \in (\hat{f}_X)^n(\emptyset)$, there is a state node Θ and a set of deferrals $d' \subseteq d(\Theta)$ such that $\Theta \vdash_{PL} \Delta$, $d' \vdash_{\Theta} d$ and $(\Theta, d') \in (\hat{f}_X)^n(\emptyset)$.*

Proof. We proceed by induction over the pair $(u_f(\Delta), u_p(\Delta))$ in lexicographic order $<_l$. If $u_f(\Delta) = 0$ and $u_p(\Delta) = 0$, then Δ is a state node so that it suffices to put $\Theta = \Delta$ and $d' = d$.

Otherwise Δ is not a state node so that at least one rule matches Δ . Let $\Sigma \in Cn(\Delta) \neq \emptyset$. Since $\Delta \in (\hat{f}_X)^n(\emptyset)$, there is a $\Gamma \in \Sigma$ with $(\Gamma, d_{\Delta \rightsquigarrow \Gamma}) \in X \cap (\hat{f}_X)^{n-1}(\emptyset) \subseteq (\hat{f}_X)^n(\emptyset)$. Also $d_{\Delta \rightsquigarrow \Gamma} \subseteq d(\Gamma)$ and since Γ is obtained from Δ as conclusion of a non-modal rule, $\Gamma \vdash_{PL} \Delta$. We note that since $\Gamma \vdash_{PL} \Delta$ and $d \subseteq d(\Delta) \subseteq \Delta$, we have $d_{\Delta \rightsquigarrow \Gamma} \vdash_{\Gamma} d$. As the non-modal rule either unfolds one unguarded fixpoint literal which then becomes guarded or removes one unguarded propositional connective from Δ , we have that $(u_f(\Gamma), u_p(\Gamma)) <_l (u_f(\Delta), u_p(\Delta))$ so that by induction we have a state node Θ and a set $d' \subseteq d(\Theta)$ with $\Theta \vdash_{PL} \Gamma$, $d' \vdash_{\Theta} d_{\Delta \rightsquigarrow \Gamma}$ and $(\Theta, d') \in (\hat{f}_X)^n(\emptyset)$. By transitivity of propositional entailment, $\Theta \vdash_{PL} \Delta$ and $d' \vdash_{\Theta} d$ so that we are done. \blacktriangleleft

► **Definition 51.** A formula ϕ is *(closed-)respected* if $\widehat{\llbracket \eta X. \psi \rrbracket} \subseteq \llbracket \eta X. \psi \rrbracket$ for each (closed) fixpoint literal $\eta X. \psi \leq \phi$. We extend the notion of pseudo-extension to sets Ψ of formulas by putting $\widehat{\llbracket \Psi \rrbracket} = \bigcap_{\psi \in \Psi} \widehat{\llbracket \psi \rrbracket}$.

► **Definition 52.** Given a sequence σ , we define the interpretation $\hat{\sigma}$ as $\hat{\sigma}(Y) = \widehat{\llbracket \sigma(Y) \rrbracket}$, for each $Y \in \mathfrak{V}$. We put $\llbracket \alpha \rrbracket \hat{\sigma} = \llbracket \alpha \rrbracket_{\hat{\sigma}}$.

► **Lemma 53.** Let ψ be a closed-respected formula. Then

- a) $\widehat{\llbracket \nu X. \psi \rrbracket} \subseteq \llbracket \nu X. \psi \rrbracket$ and
- b) $\widehat{\llbracket \mu X. \psi \rrbracket} \subseteq \llbracket \mu X. \psi \rrbracket$.

Proof. For a), we note that $\llbracket \nu X. \psi \rrbracket = \nu \llbracket \psi \rrbracket_X$. Hence we proceed by coinduction, i.e. we show that $\widehat{\llbracket \nu X. \psi \rrbracket} \subseteq \llbracket \psi \rrbracket_X \widehat{\llbracket \nu X. \psi \rrbracket} = \llbracket \psi \rrbracket(\widehat{\llbracket \nu X. \psi \rrbracket})$. We have $\widehat{\llbracket \nu X. \psi \rrbracket} = \llbracket \psi[X \mapsto \widehat{\llbracket \nu X. \psi \rrbracket}] \rrbracket = \llbracket \psi(\widehat{\llbracket \nu X. \psi \rrbracket}) \rrbracket$. As $\psi <_f \nu X. \psi$, Lemma 54 finishes the case. For b), notice that

$$\widehat{\llbracket \mu X. \psi \rrbracket} = \llbracket \psi[X \mapsto \widehat{\llbracket \mu X. \psi \rrbracket}] \rrbracket = \llbracket \psi(\widehat{\llbracket \mu X. \psi \rrbracket}) \rrbracket$$

and that $(\psi, (\mu X. \psi))$ is $\mu X. \psi$ -deferral. Also $\llbracket \mu X. \psi \rrbracket = \llbracket \psi(\mu X. \psi) \rrbracket$. Let $v \in \llbracket \psi(\mu X. \psi) \rrbracket$ and note that by definition of sufficiency (Definition 21), $d(l(v)) \vdash_{l(v)} \psi(\mu X. \psi)$. Since $v \in W$ and since L is a timed-out tableau, we have $v \in to(d(\Delta), n)$ for some n . By Lemma 55, $v \in \llbracket \psi(\mu X. \psi) \rrbracket$, as required. \blacktriangleleft

► **Lemma 54.** For all $\sigma = [X_1 \mapsto \chi_1]; \dots; [X_n \mapsto \chi_n]$ and all closed-respected formulas ψ with $\psi <_f \chi_1$,

$$\widehat{\llbracket \psi \sigma \rrbracket} \subseteq \llbracket \psi \rrbracket \hat{\sigma}.$$

Proof. We proceed by induction over ψ . If $\psi = \perp$, $\psi = \top$, $\psi = p$ or $\psi = \neg p$, for $p \in P$, then ψ is closed and $\widehat{\llbracket \psi \rrbracket} = \llbracket \psi \rrbracket$ so that $\widehat{\llbracket \psi \sigma \rrbracket} = \widehat{\llbracket \psi \rrbracket} = \llbracket \psi \rrbracket = \llbracket \psi \rrbracket \hat{\sigma}$. If $\psi = X$, then $\widehat{\llbracket X \sigma \rrbracket} = \widehat{\llbracket \sigma(X) \rrbracket} = \hat{\sigma}(X) = \llbracket X \rrbracket_{\hat{\sigma}} = \llbracket X \rrbracket \hat{\sigma}$. If $\psi = \psi_1 \wedge \psi_2$, then $\llbracket (\psi_1 \wedge \psi_2) \sigma \rrbracket = \llbracket \psi_1 \sigma \wedge \psi_2 \sigma \rrbracket = \llbracket \psi_1 \sigma \rrbracket \cap \llbracket \psi_2 \sigma \rrbracket \subseteq \llbracket \psi_1 \rrbracket \hat{\sigma} \cap \llbracket \psi_2 \rrbracket \hat{\sigma} = \llbracket \psi_1 \wedge \psi_2 \rrbracket \hat{\sigma}$, where the inclusion is by the induction hypothesis. The case for disjunction is analogous. If $\psi = \langle a \rangle \psi_1$, then

$$\begin{aligned} \llbracket \langle a \rangle \psi_1 \sigma \rrbracket &\subseteq \{v \mid \exists w \in R_a(v). w \in \llbracket \psi_1 \sigma \rrbracket\} \\ &\subseteq \{v \mid \exists w \in R_a(v). w \in \llbracket \psi_1 \rrbracket \hat{\sigma}\} \\ &= \llbracket \langle a \rangle \psi_1 \rrbracket \hat{\sigma}, \end{aligned}$$

where the second inclusion follows from the induction hypothesis and the first inclusion holds as follows: Let $v \in \llbracket \langle a \rangle \psi_i \sigma \rrbracket$ and let $R_a(v) = \{w_1, \dots, w_m\}$. There is a $\langle a \rangle$ -rule that matches $\langle a \rangle \psi_i \sigma$ as well as a number of $[a]$ -literals from $l(v)$, i.e. that matches

$\Gamma, [a]\phi_1, \dots, [a]\phi_n, \langle a \rangle \psi_i \sigma = l(v)$ and has the conclusion $\{\{\phi_1, \dots, \phi_n, \psi_i \sigma\}\} = \Sigma_j$ for some $1 \leq j \leq m$. As $w_j \in L(v)$ and L is a timed-out tableau, $w_j \in \llbracket \psi_i \sigma \rrbracket$, as required. If $\psi = [a]\psi_1$, then

$$\begin{aligned} \llbracket ([a]\psi_1)\sigma \rrbracket &\subseteq \{v \mid \forall w \in R_a(v). w \in \llbracket \psi_1 \sigma \rrbracket\} \\ &\subseteq \{v \mid \forall w \in R_a(v). w \in \llbracket \psi_1 \rrbracket \hat{\sigma}\} \\ &= \llbracket [a]\psi_1 \rrbracket \hat{\sigma}, \end{aligned}$$

where the second inclusion follows from the induction hypothesis and the first inclusion holds as follows: Let $v \in \llbracket ([a]\psi_1)\sigma \rrbracket$ and let $R_a(v) = \{w_1, \dots, w_m\}$. Either there is no $\langle a \rangle$ -rule that matches v in which case $R_a(v) = \emptyset$ and we are done; or there is a $\langle a \rangle$ -rule matching $[a]\psi_i \sigma$ as well as a number of other $[a]$ -literals and one $\langle a \rangle$ -literal from $l(v)$, i.e. matching $\Gamma, [a]\phi_1, \dots, [a]\phi_n, [a]\psi_i \sigma, \langle a \rangle \phi = l(v)$ and having $\{\{\phi_1, \dots, \phi_n, \psi_i \sigma, \phi\}\} = \Sigma_j$ as conclusion, for some $1 \leq j \leq m$. As $w_j \in L(v)$ and L is a timed-out tableau, $w_j \in \llbracket \psi_i \sigma \rrbracket$, as required. If $\psi = \nu Y. \psi_1$, then

$$\begin{aligned} \llbracket (\nu Y. \psi_1)\sigma \rrbracket &= \llbracket (\psi_1[Y \mapsto \nu Y. \psi_1])\sigma \rrbracket \\ &= \llbracket \psi_1([Y \mapsto \nu Y. \psi_1]; \sigma) \rrbracket \\ &\subseteq \llbracket \psi_1 \rrbracket ([Y \mapsto \nu Y. \psi_1]; \sigma), \end{aligned}$$

where the inclusion is by the induction hypothesis, showing by coinduction that $\llbracket (\nu Y. \psi_1)\sigma \rrbracket \subseteq \llbracket \nu Y. \psi_1 \rrbracket \hat{\sigma}$, as required. If $\psi = \mu Y. \psi_1$, $\mu Y. \psi_1$ is closed so that $\llbracket \mu Y. \psi_1 \rrbracket = \llbracket \mu Y. \psi_1 \rrbracket \subseteq \llbracket \mu Y. \psi_1 \rrbracket \hat{\sigma} = \llbracket \mu Y. \psi_1 \rrbracket \hat{\sigma}$, where the inclusion is by assumption. \blacktriangleleft

► **Lemma 55.** *For all closed-respected deferrals δ , all focused nodes $v \in D$, all sets of deferrals $d \subseteq d(l(v))$ and all $n \geq 0$,*

$$\text{if } d \vdash_{l(v)} \delta \text{ and } v \in to(d, n), \text{ then } v \in \llbracket \delta \rrbracket.$$

Let $\delta = \alpha\sigma$ and recall that the assumption of the lemma implies that $v \in \llbracket \alpha\sigma \rrbracket$. We proceed by induction over the triple $(n, m := u_f(\alpha\sigma), \alpha)$ in lexicographic order $<_l$. Let $[X \mapsto \mu X. \psi]$ and $[X_n \mapsto \theta]$ be the first and last substitutions in σ , respectively. If $d = \emptyset$, then $\vdash_{l(v)} \alpha\sigma$ so that we cannot reach the modal cases in the upcoming case distinction – otherwise $\vdash_{l(v)} \langle a \rangle \alpha_1 \sigma$ iff $N(l(v), \theta) \vdash_{PL} \langle a \rangle \alpha_1 \sigma$ iff $\langle a \rangle \alpha_1 \sigma \in N(l(v), \theta)$, where $(\langle a \rangle \alpha_1, \sigma)$ is a θ -deferral, which is a contradiction since $N(l(v), \theta)$ denotes the set of formulas that are *not* induced by a θ -deferral. Analogously, the same holds for $[a]\alpha_1 \sigma$. If $u_f(\alpha\sigma) = 0$, the case that $\alpha = X$ may not occur. Recall moreover that δ is closed-respected, and hence in particular all closed subformulas of α are respected.

- As (α, σ) is a deferral, α is open so that $\alpha \neq \perp$, $\alpha \neq \top$, $\alpha \neq p$ and $\alpha \neq \neg p$, for $p \in P$.
- If $\alpha = Y$, then let $[Y \mapsto \chi_i]$ with $\chi_i = \mu Y. \psi_i$ be the first substitution in σ that touches Y , so that $\sigma = [X_1 \mapsto \chi_1]; \dots; [Y \mapsto \chi_i]; [X_{i+1} \mapsto \chi_{i+1}]; \dots; [X_n \mapsto \chi_n]$, and $v \in \llbracket \psi_i \sigma' \rrbracket$, where $\sigma' = [Y \mapsto \chi_i]; [X_{i+1} \mapsto \chi_{i+1}]; \dots; [X_n \mapsto \chi_n]$ and $d \vdash_{l(v)} \psi_i \sigma'$; also $u_f(\psi_i \sigma') < m$, (ψ_i, σ') is a deferral and $v \in to(d, n)$. By the induction hypothesis, $v \in \llbracket \psi_i \sigma' \rrbracket = \llbracket Y \sigma \rrbracket$.
- If $\alpha = \langle a \rangle \alpha_1$, then we have to show that there is a w such that $v R_a w$ and $w \in \llbracket \alpha_1 \sigma \rrbracket$. Recall that $v \in \llbracket \langle a \rangle \alpha_1 \sigma \rrbracket$ and let $R_a(v) = \{w_1, \dots, w_m\} \subseteq L(v)$. There is a $\langle a \rangle$ -rule that matches $\langle a \rangle \alpha_1 \sigma$ as well as a number of $[a]$ -literals from $l(v)$, i.e. that matches $\Gamma, [a]\phi_1, \dots, [a]\phi_n, \langle a \rangle \alpha_1 \sigma = l(v)$ and has the conclusion $\{\{\phi_1, \dots, \phi_n, \alpha_1 \sigma\}\} = \Sigma_i$ for some i . As $w_i \in L(v)$ and L is a timed-out tableau, $w_i \in \llbracket \{\phi_1, \dots, \phi_n, \alpha_1 \sigma\} \rrbracket \subseteq \llbracket \alpha_1 \sigma \rrbracket$.

We abbreviate w_i by w and note that we are done if $w \in \llbracket \alpha_1 \sigma \rrbracket$. Since L is the relation of a timed-out tableau, $w \in \text{to}(d', n-1)$ where $d' \subseteq d(w)$ and $d' \vdash_w d_{l(v) \rightsquigarrow \Gamma}$. If $(\alpha_1, \sigma) \in d_{l(v) \rightsquigarrow \Gamma}$, we have $d' \vdash_{l(w)} \alpha_1 \sigma$. Otherwise $\vdash_{l(w)} \alpha_1 \sigma$ and hence $d' \vdash_{l(w)} \alpha_1 \sigma$ as well; also (α_1, σ) is a deferral. As $(n-1, u_f(\alpha_1 \sigma), \alpha_1) <_l (n, m, \alpha)$, the induction hypothesis implies $w \in \llbracket \alpha_1 \sigma \rrbracket$, as required.

- The case for $[a]$ is analogous (cf. the proof of Lemma 54).
- If $\alpha = \alpha_1 \wedge \alpha_2$, then $v \in \widehat{\llbracket \alpha_1 \sigma \rrbracket} \cap \widehat{\llbracket \alpha_2 \sigma \rrbracket}$. For any $i \in \{1, 2\}$ for which (α_i, σ) is deferral, the induction hypothesis implies – since $d \vdash_{l(v)} \alpha_i \sigma$, $v \in \text{to}(d, n)$, $u_f(\alpha_i \sigma) \leq m$ and $(n, m, \alpha_i) <_l (n, m, \alpha)$ – $v \in \llbracket \alpha \sigma \rrbracket$. If α_i is closed, $v \in \widehat{\llbracket \alpha_i \sigma \rrbracket} = \widehat{\llbracket \alpha_1 \rrbracket}$ and since α and hence also α_1 is closed-respected, $v \in \llbracket \alpha_1 \rrbracket = \llbracket \alpha_1 \sigma \rrbracket$.
- The case for $\alpha = \alpha_1 \vee \alpha_2$ is analogous to the previous case.
- If $\alpha = \nu Y. \alpha_1$, then $\nu Y. \alpha_1$ is – since fixpoint literals are alternation-free – closed so that the induction hypothesis is not needed as we have $v \in \llbracket (\nu Y. \alpha_1) \sigma \rrbracket = \llbracket \nu Y. \alpha_1 \rrbracket$ and since α is closed-respected, $v \in \llbracket \nu Y. \alpha_1 \rrbracket = \llbracket (\nu Y. \alpha_1) \sigma \rrbracket$, as required.
- If $\alpha = \mu Y. \alpha_1$, then $v \in \llbracket (\mu Y. \alpha_1) \sigma \rrbracket = \llbracket \alpha_1 (\mu Y. \alpha_1, \sigma) \rrbracket$ and $d \vdash_{l(v)} (\mu Y. \alpha_1) \sigma$ iff $d \vdash_{l(v)} \alpha_1 (\mu Y. \alpha_1, \sigma)$. Also $(\alpha_1, (\mu Y. \alpha_1, \sigma))$ is deferral, $v \in \text{to}(d, n)$ and α_1 is closed-respected so that the induction hypothesis implies $v \in \llbracket \alpha_1 (\mu Y. \alpha_1, \sigma) \rrbracket = \llbracket (\mu Y. \alpha_1) \sigma \rrbracket$, as required.

This finishes the proof. \blacktriangleleft

► **Lemma 56.** *All closed fixpoint literals are respected.*

Proof. Let $\eta X. \psi$ be a closed fixpoint literal. We proceed by induction over the depth of nesting of closed fixpoint literals $n = \text{cfd}(\eta X. \psi)$ in $\eta X. \psi$. If $n = 1$, then ψ contains no closed fixpoint literals and is hence closed-respected so that if $\eta = \mu$, case a) and if $\eta = \nu$, case b) of Lemma 53 applies and finishes the case. If $n > 1$, then any closed fixpoint literal $\eta Y. \phi \leq \psi$ has a depth of nesting of closed fixpoint literals less than n and is respected by induction. Thus ψ is closed-respected so that Lemma 53 finishes the proof. \blacktriangleleft

Proof of Lemma 25: We proceed by induction over ψ . If $\psi = \perp$, $\psi = \top$, $\psi = p$ or $\psi = \neg p$, for $p \in P$, then $\widehat{\llbracket \psi \rrbracket} = \llbracket \psi \rrbracket$ by definition. For the propositional connectives, the inductive step is straightforward. If $\psi = \langle a \rangle \psi_1$, then note there is for any state $v \in \widehat{\llbracket \langle a \rangle \psi_1 \rrbracket}$, and any focused node (Δ, d) , a rule

$$(\Gamma, [a]\phi_1, \dots, [a]\phi_n, \langle a \rangle \psi_1 / \phi_1, \dots, \phi_n, \psi_1)$$

matching Δ , i.e. with $\Delta = \Gamma, [a]\phi_1, \dots, [a]\phi_n, \langle a \rangle \psi_1$. As we operate in a Kripke structure over a timed-out tableau, $(\Delta, d) \in \text{to}(d(\Delta), m)$ so that there is a focused node $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in L(\Delta, d)$ with $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in R_a(\Delta, d)$ and $(\Theta, d_{\Delta \rightsquigarrow \Theta}) \in \llbracket \{\phi_1, \dots, \phi_n, \psi_1\} \rrbracket \subseteq \llbracket \psi_1 \rrbracket$. The induction hypothesis finishes the case. The case where $\psi = [a]\psi_1$ is analogous. If $\psi = \eta X. \psi_1$, then Lemma 56 finishes the case. \blacktriangleleft

Proof of Theorem 28: The algorithm terminates and as we have seen, it is sound and complete, thus it decides the problem. Let $n = |\phi_0|$ where ϕ_0 denotes the input formula. The algorithm consists of a loop which is repeated at most $a := 2^n$ times since any of the at most a nodes from \mathbf{N} has been expanded after at most a expansion steps. The body of the loop consists of one expansion step and one optional propagation step. Since we are interested in worst-case performance of the algorithm, we ignore the optional propagation step. Since modal, propositional and fixpoint literal expansion is implementable in EXPTIME, the expansion step runs in EXPTIME. We convince ourselves that the propagation step runs in EXPTIME as well, which intuitively follows from the fact that propagation computes

fixpoints over G where $|G| \leq 3^n$. We consider the computation of the set E_G and note that analysis of the computation of A_G is analogous. As $E_G = \nu(X \mapsto \mu(\hat{f}_X)) = (X \mapsto \mu(\hat{f}_X))^m(\mathbf{C}_G)$ for some $m \leq 3^n$, the computation consists of at most 3^n computations of $\mu(\hat{f}_X)$, each for some $X \subseteq \mathbf{C}_G$. A single computation of $\mu(\hat{f}_X) = (\hat{f}_X)^o(\emptyset)$ for some $o \leq 3^n$ consists of at most 3^n computations of $\hat{f}_X(Y)$, each for some $Y \subseteq \mathbf{C}_G$. The computation of $\hat{f}_X(Y)$ checks for each $(\Gamma, d) \in \mathbf{C}_G$ whether there is a conclusion $(\Theta, d_{\Gamma \rightsquigarrow \Theta}) \in X \cap Y$ (or $(\Theta, d(\Theta)) \in X$, if $d = \emptyset$) for each rule that matches Γ . Propagation thus runs in time at most $(3^n)^c = 3^{c \cdot n}$ for some constant c , and, therefore, the algorithm runs in EXPTIME; modal expansion can be implemented in time $2^{\mathcal{O}(n)}$ in the relational case so that the runtime of the algorithm is bounded by $2^{\mathcal{O}(n)}$. \blacktriangleleft

A.4 Details on New Benchmark Formulas in Section 5

We define the formulas $c(x, n)$ by putting $c(x, n) := c_n(x, n)$, where $c_n(x, i)$ is defined recursively as

$$\begin{aligned} c_n(x, i) &= (\neg x_{n-i} \wedge AX \ x_{n-i} \wedge \psi_n(x, i-1)) \vee (x_{n-i} \wedge AX \ \neg x_{n-i} \wedge c_n(x, i-1)) \\ \psi_n(x, i) &= (\neg x_{n-i} \vee AX \ x_{n-i}) \wedge (x_{n-i} \vee AX \ \neg x_{n-i}) \wedge \psi_n(x, i-1). \end{aligned}$$